MASTER EN SCIENCES MATHÉMATIQUES

Implémentation d'un modèle de prédiction de dynamique des contenus sur les réseaux sociaux

Petit, Pierre

Award date:
2018

Awarding institution:
Universite de Namur

UNIVERSITE DE NAMUR

Faculté des Sciences

IMPLÉMENTATION D'UN MODÈLE DE PRÉDICTION DE DYNAMIQUE DES CONTENUS SUR LES RÉSEAUX SOCIAUX

Mémoire présenté pour l'obtention

du grade académique de master en mathématiques à finalité spécialisée en data science

Pierre PETIT

Promoteur : Renaud LAMBIOTTE

Septembre 2018

Remerciements

Dans un premier temps, je souhaite remercier mes parents, Michaël et Sabine, pour leur soutien presque inconditionnel tout au long de mon parcours universitaire ainsi que pour leurs implications respectives dans ce mémoire. Je voudrais également remercier mes collègues de classes sans lesquels ce mémoire n'aurait pas été le même, de même que toutes ces heures passées sur les bancs du département de mathématiques. Un tout grand merci également aux personnes qui ont représenté un soutien moral non négligeable tout au long de la réalisation de ce manuscrit.

Je souhaite également remercier mon promoteur Renaud Lambiotte pour son suivi, nos entres-vues constructives et ses bonnes idées tout au long du travail accompli. Merci à Juan Cabrera d'avoir consacré une partie de son précieux temps pour m'aider à régler tous les petits inconvénients informatiques que j'ai pu rencontrer lors de mes diverses implémentations.

Finalement, un merci tout particulier à ma copine, Clémentine, qui en plus de son soutien constant dans ce travail, a su, même dans les moments de crise, m'aider à garder la tête froide.

"*Ne vous souciez pas d'être meilleur que vos contemporains ou vos prédécesseurs, essayez d'être meilleur que vous-même.*"
William Faulkner

Abstract

Social medias have acquired an important part of our life. Nowadays, almost everyone uses online social networks for entertainment or professional reasons. Understanding how the information spreads on these networks is a goal that's important from many points of views such as marketing and territory security. Our objective through this manuscript is to develop a way to predict the popularity of a given content on Twitter. To do so, we used a variation of Hawkes process and Fourier's series theory just as much as machine learning concepts and we gathered empirical data from Twitter to calibrate our model. What stands out from this work is our ability to predict with a 48 hour of empirical data observation with a certain precision the reuse of a given content on social networks.

Les réseaux sociaux font désormais part intégrante de la société. Des plus petits aux plus grands, tous ont une place dans ce que l'on pourrait appeler la communication 2.0. Dès lors, les enjeux liés à ces réseaux sociaux et aux plateformes qui leurs sont liées sont grandissants que ce soit dans des objectifs commerciaux, des objectifs de sécurité, et bien d'autres. Nous allons au cours de ce travail présenter une méthode visant à prédire la popularité ou dynamique d'un contenu donné sur les réseaux sociaux. Pour ce faire, nous utiliserons des outils divers et variés tels que les processus de Hawkes pour définir le processus de réutilisation d'un contenu, les séries de Fourier afin de modéliser la viralité induite par un contenu et des techniques de machine learning afin d'optimiser cette modélisation. Ceci nous permettra de proposer une prédiction de popularité des contenus possédant une certaine précision en nous basant sur 48 heures de données empiriques.

Table des matières

Introduction **1**

1 Réseaux sociaux, contenus et processus ponctuels **3**

1.1 Réseaux sociaux, médias sociaux et connexions 3

1.1.1 Réseaux sociaux et médias sociaux 3

1.1.2 Connexions sur le réseau 6

1.2 Caractéristiques des contenus 9

1.2.1 Caractéristiques indirectement liées au contenu et à la cascade . . 9

1.2.2 Caractéristiques propres au contenu 9

1.2.3 Contenu particulier: le hashtag 10

1.3 Processus ponctuels auto-excités 11

1.3.1 Processus ponctuel 12

2 Modèle de prédiction **17**

2.1 Modèle général 17

2.2 Taux d'infection $p(t)$ 20

2.2.1 Méthode des moyennes glissantes 20

2.2.2 Estimation du taux d'infection par la méthode des fenêtres glissantes 22

2.2.3 Modélisation et optimisation des paramètres 23

2.3 Outils numériques 25

2.3.1 Noyau mémoriel: forme, estimation et paramètres 25

2.3.2 Résolution de l'équation intégrale linéaire de Volterra du second type . . . 26
2.3.3 Ajustement de courbes et méthode de Levenberg-Marquardt . . . 27
2.4 Synthèse . . . 29

3 Données et traitement **31**
3.1 L'API Twitter et le module Python Tweepy . . . 31
3.2 Pré-traitement des données . . . 32
3.3 Présentation des données . . . 34
3.3.1 Dynamiques particulières . . . 36

4 Modélisation de l'infectiosité **42**
4.1 Estimateurs du maximum de vraisemblance par les fenêtres glissantes . . . 43
4.2 Modélisation par les séries de Fourier . . . 45
4.3 Overfitting, underfitting et nombre idéal de paramètres . . . 48
4.3.1 Utilisation des concepts machine learning . . . 50

5 Prédiction de dynamique des hashtags **58**
5.1 Résumé des résultats . . . 63

Conclusion **66**

Références **67**

Annexes **72**
A Procédure de récupération de données sur Twitter . . . 73
B Procédures de prétraitement et d'affichage des données (Python) . . . 74
C Procédure d'affichage des moyennes glissantes (Python) . . . 76
D Procédure d'affichage de l'infectiosité (Python) . . . 77
E Procédure d'ajustement de l'infectiosité (Python) . . . 78
F Procédure d'affichage de la prédiction (Python) . . . 82

G Codes C 83
G.1 Header et sous-routines 83
G.2 Estimation de l'infectiosité 85
G.3 Prédiction d'activité 88

Introduction

De nos jours, les médias sociaux sont omniprésents dans la société. Des plus jeunes aux plus âgés, la population mondiale par le biais d'internet et de ces médias sociaux est en contact constant et l'information est relayée de manière extrêmement rapide. Nous pouvons même, dans le cas des médias sociaux actuels considérer cet éparpillement et cette transmission d'information comme une certaine viralité, ou comme un phénomène de contagion. Les enjeux de l'étude de ces réseaux et des activités qui s'y développent sont multiples, pouvant d'autant être orienté vers des aspects de marketing (recherche de visibilité, d'expansion, gain de "followers", etc.), vers des aspects professionnels (échange de documents, d'idées, discussions instantanées), des aspects humoristiques et de détente mais également des aspects de sécurité (par exemple en détectant et/ou en traquant des activités terroristes).

Le paysage des médias sociaux ne cesse de s'élargir d'année en année et le nombre d'utilisateurs de ceux-ci croît lui aussi d'année en année [1]. Ces médias sociaux, souvent par abus de langage appelés réseaux sociaux, proposent différentes fonctionnalités de partage de contenu comme par exemple des photos, vidéos, des états d'esprit mais aussi des articles journalistiques, etc. Malgré les bienfaits apportés par l'ensemble de ces moyens de transmission d'information, il est nécessaire que les utilisateurs de ces médias sociaux conservent un esprit critique sur ce qui peut les atteindre sur ceux-ci afin d'évaluer la véracité des propos qu'ils peuvent véhiculer.

Ce manuscrit s'intéresse à un contenu bien particulier d'un média social bien particulier: le hashtag de Twitter. Nous allons, tout au long de ce manuscrit, tenter de développer une méthode, un modèle, cherchant à pronostiquer la popularité d'un contenu sur un média social donné. Notre modèle, bien que construit sur des données liées aux hashtags de Twitter, se veut générique et adaptable à tous les médias sociaux proposant des interactions de type "follower-followee" ou des relations dites sous forme d'abonnement.

Afin de parfaire notre objectif, nous proposerons dans un premier temps quelques définitions et énoncés de concepts liés aux réseaux sociaux. Nous nous pencherons en particulier sur les différentes interactions possibles sur les réseaux et nous nous attarderons sur le contenu qui a fait l'objet de notre étude, le hashtag. Nous présenterons également brièvement le concept de processus ponctuels sur lequel se base notre modèle. Une fois les processus ponctuels introduits, nous pourrons proposer le modèle sur lequel se base notre canevas de prédiction d'activité et nous pourrons également détailler les outils numériques permettant de mettre en place la prédiction d'activité. La prédiction de notre modèle se basant sur des données empiriques, nous nous attarderons un temps sur la méthodologie mise en place afin de construire une base de données composée de hashtags de Twitter ainsi que sur les dynamiques qui peuvent se dégager des données récoltées. Ceci étant fait, nous nous pencherons sur la méthodologie adoptée pour modéliser l'infectiosité de notre modèle en introduisant un avis critique sur la manière de modéliser celle-ci et en proposant successivement plusieurs améliorations de modélisation. Finalement, nous posséderons tous les outils nécessaires pour tester notre modèle et présenterons par conséquent les résultats que nous prodigue celui-ci.

Chapitre 1

Réseaux sociaux, contenus et processus ponctuels

Dans ce premier chapitre nous allons nous atteler à définir ce que nous entendons par l'expression *"média social"*, que nous aurons également l'habitude de qualifier de *réseau social*. Nous mettrons en évidence par le biais de cette définition et de celle de "réseau social" la différence entre ces deux-ci mais adopterons l'abus de langage consistant à appeler un média social "réseau social" comme il en est de coutume dans la langue française courante. Nous définirons également les interactions qui peuvent se manifester sur différents médias sociaux, les différentes caractéristiques des contenus de ceux-ci ainsi que les processus ponctuels auto-excités qui nous serviront à définir notre modèle.

1.1 Réseaux sociaux, médias sociaux et connexions

Nous allons maintenant brièvement décrire le cadre d'étude de ce manuscrit, les données utilisées provenant du média social Twitter.

1.1.1 Réseaux sociaux et médias sociaux

Dans un premier temps, donnons une définition de ce qu'est un réseau social. Dans cette optique, nous allons tout d'abord observer plusieurs définitions de ce qu'est un réseau social au travers de plusieurs sites internet.

≪ *Mode d'interactions sociales qui facilite la création et l'échange d'informations et de contenus entre des individus et entre des individus et des organisations. Désigne aussi les plateformes qui rendent ces interactions possibles (telles que Facebook, Twitter, etc.)* ≫ [2]

≪ *Site internet qui permet aux internautes de se créer une page personnelle afin de partager et d'échanger des informations, des photos ou des vidéos avec leur communauté d'amis et leur réseau de connaissances.* ≫ [3]

≪ *Un réseau social désigne un ensemble de personnes réunies par un lien social.* ≫ [4]

La première de ces définitions nous permet de mettre en exergue l'abus de langage commun consistant à appeler les plateformes qui permettent les interactions sociales décrites dans cette même définition des réseaux sociaux. Les plateformes qui laissent l'opportunité à ce genre d'interactions portent en réalité le nom de médias sociaux. Comme nous le fait remarquer la seconde définition, cet abus de langage est répandu et la différence entre réseau social et médial social n'est généralement pas faite. Afin de marquer cette différence, nous proposons nos propres définitions de réseau social et de médial social, tout en rappelant que à l'avenir, nous nous accorderons le privilège d'abus de langage énoncé ci-avant.

Définition 1. *Réseau social*

Un réseau social est une communauté de personnes reliées par des liens sociaux. Cette communauté permet de faciliter les échanges de contenus entre les membres du réseau, ceux-ci pouvant être des personnes physiques comme des entités morales (entreprises, marques, etc.).

Notons que l'utilisation du mot réseau est pertinente, celle-ci permettant de considérer chacun des membres de la communauté comme un nœud et chacun des liens sociaux comme un arc, créant dans une idée plus mathématique un réseau de liens sociaux, nous rappelant la notion de graphe composé de sommets ou nœuds et d'arêtes ou arcs.

Il nous revient maintenant de définir ce que l'on entend par média social. Et la tâche est loin d'être aisée, car à l'heure actuelle, il existe une multitude de médias sociaux possédant tous des caractéristiques bien différentes. Nous nous efforcerons de donner une définition générique de ces médias sociaux et présenterons très brièvement certains d'entre eux.

Définition 2. *Média social*

Un média social est une plateforme informatique facilitant l'échange d'informations, de données, de contenus entre les membres d'un réseau social.

Les médias sociaux tels que nous les connaissons aujourd'hui sont des sites internets ou des applications mobiles. Un exemple de média social est la plateforme WebCampus de l'Université de Namur [5]. WebCampus permet bien des échanges entre plusieurs intervenants d'un réseau social (ici, l'ensemble des élèves qui suivent un cours bien particulier et le ou les professeur(s) responsables dudit cours) et facilite ces échanges (possibilité de remettre ses travaux en ligne plutôt que de devoir les imprimer, création de forums de discussion, mise à disposition de supports de cours du professeur vers les étudiants, système de prise de rendez-vous, etc.). Ces caractéristiques permettent d'affirmer, selon la définition de média social que nous avons donnée, que WebCampus est un média social.
Nous allons maintenant lister quelques uns des médias sociaux les plus populaires et décrire brièvement quelques unes des caractéristiques qui font de ceux-ci des médias sociaux.

1. **Facebook:** média social permettant de diffuser du contenu (photo, vidéo, URL, texte, etc.) et de le partager avec les utilisateurs du réseau de manière plus ou moins restreinte (allant de la diffusion privée du contenu de l'utilisateur envers lui-même jusqu'à la diffusion totalement publique du contenu). Présence de personnes physiques et d'entités morales, présentées sous forme de "Pages". Les utilisateurs peuvent réagir aux contenus de différentes manières (aimer du contenu, commenter du contenu, partager du contenu).

2. **Twitter:** média social permettant de diffuser du contenu (photo, vidéo, URL, texte, etc.) et de le partager avec les utilisateurs du réseau. La distinction entre une personne physique et une entité morale n'est pas réalisée au sein du réseau. Les publications des utilisateurs sont soumises à une limite en taille (280 caractères).

3. **YouTube:** média social hébergeant la plus grande base de données de vidéos au monde [6]. Permet de diffuser du contenu vidéo. Désormais, les comptes sont liés aux comptes Google des utilisateurs.

4. **Instagram:** média social permettant de diffuser du contenu (majoritairement photo et vidéo) et de le partager avec les utilisateurs du réseau.

5. **LinkedIn:** média social à but principalement professionnel, orienté vers le business et l'emploi. Permet aux utilisateurs de créer un profil prenant une forme de type Curriculum Vitae.

6. **Snapchat:** média social permettant de diffuser du contenu visuel et audio-visuel (photo ou vidéos) de courte durée (une à dix secondes) auquel peut être ajouté un texte court superposé au contenu audio-visuel ou visuel.

Comme annoncé précédemment, l'abus de langage d'appeler un média social "réseau social" est fréquent. Nous nous permettrons également cet écart et dès lors, pour la suite de ce manuscrit et sauf indication contraire, les termes réseau social et réseaux sociaux feront désormais référence aux termes média social et médias sociaux respectivement.

1.1.2 Connexions sur le réseau

Cette première définition étant maintenant réalisée, nous pouvons nous lancer dans une description plus précise des interactions possibles entre les utilisateurs du média social. Il est important de définir ces différentes interactions car celles-ci représentent une donnée importante de la suite de notre travail. Nous allons en effet tenter par la suite de qualifier, de quantifier, d'estimer, de prévoir ces interactions, et plus précisément nous allons nous intéresser à la popularité que peut acquérir un contenu sur un média social, et c'est de manière plus précise la popularité de ce contenu que nous allons tenter de quantifier, d'estimer et de prévoir.

Notre cadre de travail nous impose de ne pas considérer certaines fonctionnalités des réseaux sociaux listés ci-dessus. Par exemple, l'une des fonctionnalités de Snapchat est d'envoyer du contenu directement à un utilisateur sur le réseau sans rendre ce contenu accessible à l'ensemble des utilisateurs du réseau. Nous éviterons par conséquent de nous pencher sur ce type de contenu. Afin de travailler dans un cadre optimal, nous ne nous intéresserons qu'au contenu étant complètement accessible à tous les utilisateurs présents sur le réseau même dans le cas où ceux-ci ne sont pas en lien direct avec l'utilisateur ayant émis le contenu. Par lien direct nous entendons ici que les deux utilisateurs (émetteur et récepteur) sont reliés par un arc sur le réseau. Pour illustrer cela, on peut prendre le cas de Facebook. Nous considérerons dans le cas du contenu présent sur Facebook uniquement celui portant l'attribut de confidentialité "Public" signifiant que ce contenu peut être atteint et vu par tous les utilisateurs du réseau (moyennant certaines exceptions comme par exemple les utilisateurs bloqués par l'émetteur). Pour

Twitter, on considérera le contenu appelé Tweet accessible à tous les utilisateurs de la communauté.

Dès lors, une publication issue de ce que nous appellerons désormais l'émetteur originel, peut apparaître dans le réseau à peu près aux yeux de n'importe quel utilisateur moyennant un chemin plus ou moins long à suivre. Dans le réseau, des utilisateurs peuvent être connectés par des liens de manière directe (être amis ou abonné sur Facebook, être abonné sur Twitter, être abonné sur YouTube, etc.) ou de manière indirecte (avoir un ami en commun sur Facebook sans être directement amis par exemple). La question se pose maintenant de savoir la manière dont un contenu peut atteindre un utilisateur du réseau n'étant pas connecté de manière directe à l'émetteur originel. Chaque réseau social possède ses propres manières permettant à un contenu de se répandre, mais ces extensions de personnes atteintes sur le réseau sont causées par des réactions de personnes elles-même exposées au contenu. Ces réactions peuvent être internes au réseau ou externes au réseau social. Des réactions internes sont par exemple le fait de commenter, aimer ou partager une publication sur Facebook ou Twitter, ou le fait de regarder une vidéo sur YouTube et ainsi d'incrémenter son nombre de vues. Tandis que des réactions externes seront des actions effectuées en dehors du réseau lui-même, par exemple en envoyant le lien d'une vidéo de YouTube par e-mail à une sélection de contacts ou en partageant une photo publiée sur Instagram directement sur Facebook.

Introduisons également la notion de cascade d'information. On définira une cascade comme le phénomène d'éparpillement d'un contenu sur le réseau. Cette définition de cascade nous permet de définir la notion de profondeur d'une cascade, liée au phénomène de partage de contenu. Cette notion de profondeur de cascade tend à mesurer l'éloignement maximal entre la publication originale et un $n^{\text{ième}}$ partage de cette publication. Chacun des nœuds se trouve à un niveau de profondeur de la cascade, qui est au minimum de un. Déterminer la profondeur réelle d'une cascade peut s'avérer être une tâche difficile[7]. Afin d'illustrer cette notion de niveau de profondeur dans une cascade et de montrer une représentation de ce que peut être le phénomène de partage, nous allons présenter deux graphes permettant de comprendre de manière plus visuelle ces concepts.

La Figure 1.1 illustre une situation entre trois protagonistes A, B et C. Dans cette configuration, A est l'émetteur originel. Dans le premier cas, B et C suivent tous deux A - ce qui signifie en terme d'abonnements évoqués ci-avant que B et C sont abonnés à A sur le réseau. Tous deux ont été exposés à la publication de A et ont décidé de partager celle-ci. Dans le second cas, B suit A, C suit B, et il est possible mais pas nécessaire que

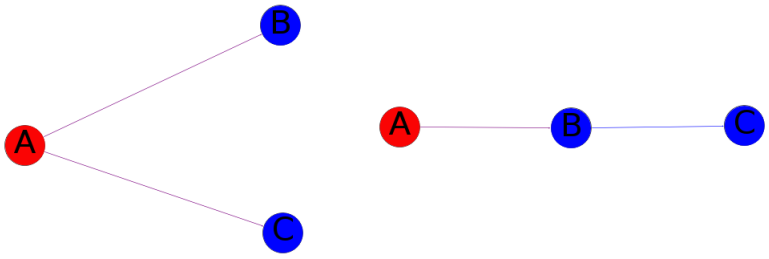


Figure 1.1: Configurations possibles d'un schéma de partage à trois protagonistes. Ici, A est l'émetteur originel et B et C partagent le contenu.

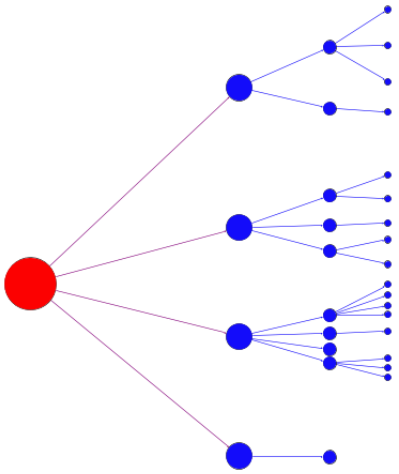


Figure 1.2: Exemple de schéma de propagation.

C suive A. B voit la publication de A et la partage. C voit la publication de B consistant au partage de la publication de A et la partage. La cascade d'information correspondant à chacun de ces cas est représentée à la Figure 1.1 par un graphe G. Dans chacune de ces situations, le nœud du graphe représentant le protagoniste B se situe à un niveau de profondeur de un dans la cascade. Le nœud du graphe représentant le protagoniste C se situe quand à lui pour la première situation à un niveau de profondeur un et à un niveau de profondeur deux pour la seconde situation.

La Figure 1.2 représente quant à elle une situation similaire faisant intervenir un nombre plus important de protagonistes et peut laisser supposer dans le cas d'utilisateurs possédant de nombreux abonnés l'ampleur que peut prendre ce genre de phénomène.

1.2 Caractéristiques des contenus

Nous allons à présent brièvement présenter les caractéristiques qui peuvent être liées aux divers contenus des réseaux sociaux. Ces caractéristiques peuvent influer sur l'évolution d'une publication sur le réseau [7, 8]. Nous allons présenter pour l'ensemble des contenus présents sur les réseaux sociaux des caractéristiques qui leurs sont propres et qui peuvent influencer la popularité et la réutilisation de tels contenus. Notre étude portant sur la viralité que peut induire un hashtag, nous dédierons également une partie de cette section à présenter ceux-ci et les caractéristiques qui leurs sont spécifiquement liées.

1.2.1 Caractéristiques indirectement liées au contenu et à la cascade

Nous allons dans un premier temps décrire des facteurs portant une influence sur le développement de la cascade qui ne sont pas directement liés au contenu de la publication ou de l'élément initiant la cascade. Nous décidons de rassembler ces différentes caractéristiques dans les quatre groupes distincts [7].

1. Caractéristiques liées à l'émetteur originel e.g. sexe de l'émetteur, âge, nombre de followers, etc.,
2. Caractéristiques liées aux utilisateurs partageant le contenu, globalement similaires aux caractéristiques de l'émetteur originel e.g. sexe, âge, nombre de followers
3. Caractéristiques structurelles du réseau e.g. nombre d'utilisateurs ayant partagé connectés directement à l'émetteur originel, profondeur maximale (estimée) de la cascade, etc.,
4. Caractéristiques temporelles e.g. temps écoulé depuis l'émission du contenu, temps moyen écoulé entre l'arrivée du contenu sur le fil d'actualité d'un utilisateur et son partage.

1.2.2 Caractéristiques propres au contenu

Nous allons maintenant nous intéresser au contenu d'une publication en lui-même et présenter les caractéristiques qui lui sont propres.

Ces caractéristiques peuvent être quantitatives tout comme qualitatives. Les caractéristiques qualitatives seront transformées de manière à devenir quantitatives, et ce pour des raisons d'implémentation pour les travaux effectués ultérieurement (prédiction, évaluation de la qualité des contenus, etc.) [7].

Nous donnons ci-après une liste non-exhaustive de caractéristiques de contenu possibles en y ajoutant parfois une courte explication:

- la composition du contenu (texte brut, photo, vidéo, url, etc.),
- la qualité du contenu,
- la langue dans laquelle le contenu est publié (anglais ou autre),
- les références visées par le contenu (lieux, personnes) sous forme d'hyperliens ou d'identifications,
- l'utilisation ou non de hashtags,
- la longueur du contenu,
- les émotions induites par le contenu (positives, négatives, neutres),
- etc.

1.2.3 Contenu particulier: le hashtag

Nous allons maintenant décrire un élément pouvant faire partie du contenu de certains réseaux sociaux et parler de ses caractéristiques propres. Un tel contenu est appelé hashtag, hash faisant référence en anglais au symbole de dièse # et tag ajoutant une référence au caractère identificatoire du hashtag. Le hashtag est un des deux méta symboles autorisés par le réseau social Twitter et présent également sur d'autres réseaux sociaux comme Facebook et Instagram. Nous allons dans un premier temps donner une définition formelle du hashtag tel que présent sur les réseaux sociaux pour ensuite énoncer les caractéristiques que ceux-ci peuvent tenir en place de contenu présent sur les réseaux sociaux [8].

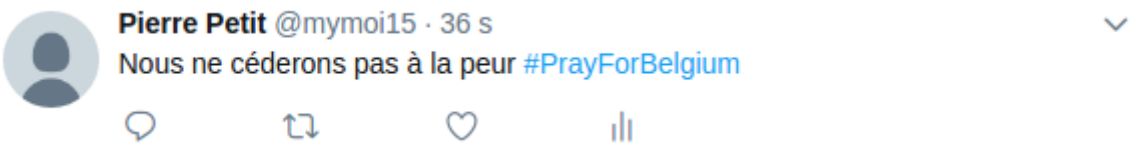


Figure 1.3: Tweet dans lequel le hashtag utilisé permet de préciser le contexte

Définition 3. *Hashtag*

Un hashtag est une suite de caractères ne contenant pas d'espaces blancs précédée par le signe #.

L'utilisation d'un hashtag permet par exemple de donner un contexte à un contenu présenté sur un réseau social. Par exemple, *#PrayForBelgium* est un hashtag. Nous pouvons imaginer ce hashtag entrant dans une publication du réseau social Twitter illustrée à la Figure 1.3. Dans un tel cas, le texte est assez vague et c'est le hashtag qui permet de donner un contexte précis au reste du contenu.

Certaines caractéristiques peuvent être directement extraites du hashtag lui-même. Nous citerons par exemple le nombre de caractères composant le hashtag, le nombre de mots présents dans le hashtag, l'orthographe utilisée, le champ lexical, la position dans le tweet, la dimension cognitive (les sentiments que peuvent provoquer le hashtag), etc. [8] La Figure 1.4 présente les différentes positions que peut prendre un hashtag dans un tweet (à la fin du texte, au sein du texte, au début du texte).

L'ensemble des caractéristiques citées sont susceptibles d'influencer la popularité d'un contenu. C'est cette popularité que nous allons tenter de quantifier, d'estimer et de prévoir.

1.3 Processus ponctuels auto-excités

Nous allons maintenant décrire ce que sont des processus ponctuels pour définir un cas particulier de ceux-ci se prêtant particulièrement bien à la prédiction de popularité de

Figure 1.4: Illustration des différentes positions possibles d'un hashtag dans un tweet

contenu sur les réseaux sociaux. De tels processus sont appelés ponctuels auto-excités temporels.

1.3.1 Processus ponctuel

Il convient avant de définir le processus ponctuel auto-excité temporel d'en définir sa base, à savoir le processus ponctuel. La rédaction de cette section se base principalement sur [9]. Afin de définir le processus ponctuel, nous devons tout d'abord nous munir d'un espace probabilisé qui permet de caractériser l'aspect aléatoire des expériences.

Soit $(\Omega, \mathcal{A}, \mathbb{P})$ un espace probabilisé et soit (χ, μ) un espace métrique complet séparable où χ est un ensemble non-vide.

Définition 4. *Configuration*

Une configuration est un ensemble de points $x = (x_1, ..., x_n, ...)$ tel que les points $x_i \in (\chi, \mu)$ pour $i \in \mathbb{N}_0$ sont issus d'une expérience aléatoire.

Définition 5. *Configuration localement finie*

Une configuration est appelée localement finie si celle-ci possède un nombre fini de points dans n'importe quel borélien borné de (χ, μ). On note N^{lf} la famille des configurations localement finies.

Ces définitions vont maintenant nous permettre de définir ce qu'est un processus ponctuel.

Définition 6. *Processus ponctuel*

Un processus ponctuel sur (χ, μ) est une application qui à une réalisation possible du hasard associe une configuration localement finie. Cette application associe donc à un élément $\omega \in (\Omega, \mathcal{A}, \mathbb{P})$ un élément $x \in N^{lf}$ dans (χ, μ). Le nombre de points dans un borélien A d'une telle configuration, noté $N(A)$ vérifie la propriété suivante: $\omega \mapsto N(A)(\omega) = N(A)$ est une variable aléatoire discrète finie pour tout borélien borné A de χ.

Définition 7. *Processus de comptage*

Un processus de comptage est un processus ponctuel $N(t), t > 0$ à valeur dans $\mathbb{N}$ qui satisfait $N(0) = 0$ et qui est une fonction étagée continue à droite d'incrément 1.[10]

Nous pouvons alors, une fois le processus ponctuel défini, préciser ce que l'on entend par processus ponctuel auto-excité.

Définition 8. *Processus ponctuel auto-excité*

Soit N un processus ponctuel. N est appelé auto-excité si $cov\,(N(s,t), N(t,u)) > 0$ avec $s < t < u$ [11]. Donc, dans un processus ponctuel auto-excité, l'apparition de points incite à plus d'apparitions futures.

Définition 9. *Processus ponctuel temporel*

Un processus ponctuel est dit temporel lorsque les réalisations de ce processus correspondent à des temps (t_i).[12]
Nous pouvons également définir $(H(u) : u \geq 0)$, l'historique des temps d'activation du processus jusqu'au temps u.

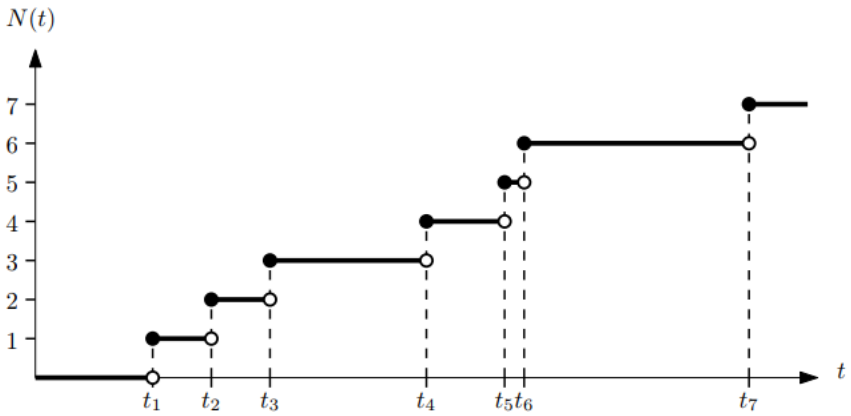


Figure 1.5: Illustation de processus ponctuel temporel et de processus de comptage. Souce: [10]

Définition 10. *Intensité du processus*

Soit un processus de comptage N et les historiques H qui lui sont associés. Si une fonction $\lambda^*(t)$ définie par

$$\lambda^*(t) = \lim_{h \downarrow 0} \frac{E(N(t+h) - N(t)|H(t))}{h}$$

existe et ne dépend que des informations de N du passé, alors λ^* est appelée intensité conditionnelle du processus.
La Figure 1.5 nous présente une illustration d'un processus ponctuel temporel et du processus de comptage qui lui est associé. Cette représentation peut par exemple modéliser l'utilisation d'un hashtag, sa première apparition sur le réseau étant liée au temps t_1 et donc la valeur du processus de comptage en t_1 étant de 1, puis sa première réutilisation en t_2 induisant donc que la valeur du processus de comptage en t_2 est de 2, etc. Les processus ponctuels auto-excités sont typiquement utilisés pour modéliser des situations du type "les riches deviennent plus riches"[13].

Processus de Hawkes

Le processus de Hawkes est un cas particulier de processus ponctuel. C'est sur ce type de processus que se basera notre modèle de prédiction décrit dans le chapitre 2.

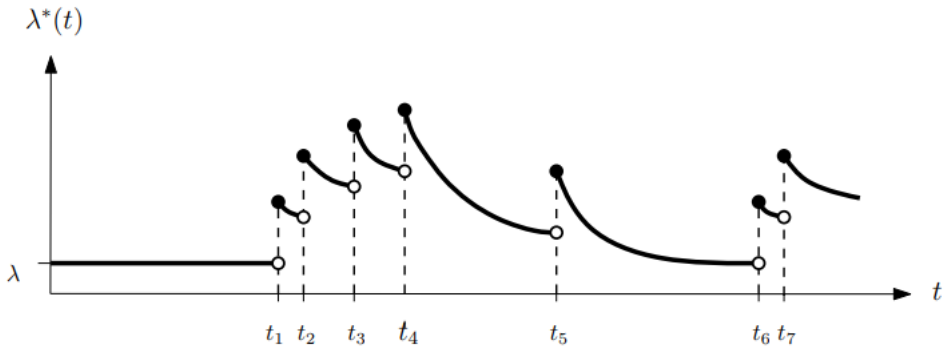


Figure 1.6: Exemple d'intensité conditionnelle d'un processus ponctuel auto-excité temporel. Source: [10]

Le processus de Hawkes est un processus ponctuel temporel auto-excité dans lequel l'intensité du processus est donnée par

$$\lambda_t = \mu_t + \sum_{i:t_i<t} \phi(t - t_i)$$

où μ_t correspond à un taux caractérisant les événements passés et ϕ est ce que l'on va appeler la mémoire du noyau ou la densité de regroupement. Le paramètre μ_t est parfois également appelé intensité de fond et la fonction ϕ fonction d'excitation.

La Figure 1.6 représente une intensité de processus pouvant par exemple être associée à un processus de Hawkes.

Modèle de prédiction basé sur les processus de Hawkes

Le modèle SEISMIC [13] est un modèle basé sur les processus ponctuels auto-excités qui va permettre de répondre à trois questions concernant un contenu présent sur un réseau social.

1. Connaissant l'historique d'occurrence d'un contenu, quel sera le nombre final d'occurrences de celui-ci ?

2. La phase dite explosive de partage du contenu s'est-elle déjà produite ou non ? Autrement dit, la cascade se trouve-t-elle dans un état sous-critique ou super-critique ?

3. Quels seront les contenus les plus partagés dans le futur ?

L'aspect qui nous intéresse est l'aspect de prédiction, donc de connaître pour un contenu donné, le nombre final d'occurrences dudit contenu. Dans le cadre de cette étude, les contenus considérés étaient les tweets et le phénomène de partage de ceux-ci, nommé retweet. Les caractéristiques de ce modèle se synthétisent par le fait que c'est un modèle génératif qui ne nécessite que l'historique de partage d'un contenu pour en prévoir l'état final, qu'il possède une complexité linéaire en le nombre d'observations et que l'algorithme construit derrière le modèle est aisément parallélisable, ainsi que par sa facilité d'interprétation.

Le modèle SEISMIC se démarque d'autres modèles basés sur les processus de Hawkes en considérant que l'intensité du processus notée λ_t dépend d'un processus stochastique p_t que nous caractérisons comme la viralité ou l'infectiosité du contenu. L'intensité du processus λ_t est dérivée du processus de Hawkes et nous obtenons comme intensité

$$\lambda_t = p_t \sum_{t_i<t, i\geq 0} n_i \phi(t - t_i)$$

où p_t est estimée à partir de l'historique connu de partage du contenu, n_i est le degré du nœud i et ϕ est appelé noyau mémoriel et permet de quantifier la réactivité humaine. Si les retweets ont été observés entre t_0 et T, le nombre final de retweets sera donné en évaluant

$$R_\infty = R_T + \int_T^\infty \lambda(s) ds$$

où R_T représente le nombre de retweets observés entre t_0 et T.

Chapitre 2

Modèle de prédiction

Dans ce chapitre nous allons décrire la méthodologie suivie afin de modéliser la dynamique des événements (réutilisation de hashtag) et ainsi d'en prédire l'activité future. La construction de ce modèle se base sur les processus ponctuels auto-excités décrits dans la section 1.3 et plus particulièrement sur les processus de Hawkes. Le canevas de construction du modèle se calque sur les étapes de construction du modèle TiDeH tel que décrit dans [14]. Ce chapitre sera séparé en quatre parties: dans un premier temps nous allons décrire la méthode générale de prédiction adoptée. Nous nous attarderons ensuite sur le paramètre d'infectiosité $p(t)$ de notre modèle, ce paramètre étant l'ingrédient phare de notre modèle. Nous nous pencherons ensuite sur les outils numériques permettant de mettre en œuvre le modèle présenté. Finalement, nous résumerons ce chapitre en réalisant la synthèse du fonctionnement de notre modèle.

2.1 Modèle général

Afin de décrire la problématique et le cadre général de la prédiction de popularité des hashtags sur les réseaux sociaux, nous devons dans un premier temps planter le décor. Notre objectif est, sur base d'une activité d'utilisation d'un hashtag connue sur un intervalle de temps fixé, de pouvoir prédire l'utilisation future dudit hashtag. Autrement dit, étant donné l'activité passée d'un hashtag, nous cherchons à en déterminer l'activité future. Nous cherchons par conséquent, pour des intervalles de temps Δt, à déterminer l'activité du hashtag sur $[t+\Delta t]$, ce qui peut se traduire par la probabilité d'observer une activité, une réutilisation de hashtag sur l'intervalle de temps concerné. Nous choisissons

de considérer que la probabilité d'obtenir une nouvelle occurrence du hashtag considéré sur l'intervalle de temps $[t + \Delta t]$ peut se traduire par

$$P([t + \Delta t]) = \lambda(t)\Delta t \tag{2.1}$$

où $\lambda(t)$ est la probabilité instantanée de réutilisation, décrite comme intensité du processus dans le cadre des processus ponctuels auto-excités de Hawkes. C'est ce λ que nous allons modéliser afin de décrire l'activité future. En se basant sur les processus de Hawkes (voir section 1.3.1) nous pouvons proposer l'équation

$$\lambda(t) = p(t) \sum_{i:t_i<t} d_i \phi(t - t_i) \tag{2.2}$$

comme intensité du processus. Dans cette proposition d'intensité du processus, nous retrouvons

- $p(t)$, le paramètre d'infectiosité ou taux d'infection ou plus simplement, l'infectiosité,
- t_i le temps correspondant à la *i-ème* occurrence de réutilisation du hashtag,
- d_i le nombre de followers de la *i-ème* personne utilisant le hashtag considéré,
- la fonction $\phi(s)$ appelée noyau mémoriel (memory kernel) ou densité de regroupement.

Le paramètre $p(t)$ est le paramètre le plus important de notre modèle et il nécessite à lui seul une modélisation complète ainsi qu'un ajustement de paramètres qui lui sont propre. La fonction $\phi(s)$ est une densité de probabilité cherchant à représenter le temps de réaction humain. Plus précisément, cette fonction va représenter l'intervalle de temps entre le moment où un utilisateur voit le hashtag dans son fil d'actualité et le moment où il décide d'utiliser le hashtag en question.
En utilisant une telle modélisation de l'intensité du processus, nous pouvons proposer d'évaluer l'activité du hashtag entre t et $t + \Delta t$ en calculant

$$\int_t^{t+\Delta t} \lambda(s) ds. \tag{2.3}$$

Cette proposition théorique est intéressante, mais non réalisable dans la pratique. En effet, intégrer λ sur $[t, t+\Delta t]$ étant donné la forme d'intensité du processus de l'équation (2.2) peut mener à deux situations distinctes. Premièrement, t correspond avec le dernier

temps de réutilisation du hashtag connu dans les données i.e. le dernier des t_i. Dans cette situation, c'est la taille de Δt qui va donner l'intervalle de prédiction (une heure, 24 heures, 48 heures, etc.). Cette situation revient dès lors à considérer une intensité du processus constante sur le temps Δt avec Δt grand. Ceci ne représente pas la réalité des réseaux sociaux et cette situation doit dès lors être écartée. La seconde situation représente quant à elle le cas où t est strictement supérieur à t_i, le temps correspondant à la dernière occurrence du hashtag dans les données. Cette situation nous permet d'évaluer une intensité du processus instantanée sur Δt. Cependant, dans cette situation, nous connaissons l'activité du hashtag jusqu'au temps t_i mais cette activité est inconnue entre t_i et t, et cependant nécessaire pour connaître $\lambda(t)$. Nous allons par conséquent proposer d'estimer le taux $\lambda(t)$ par $\hat{\lambda}(t)$ qui se traduit comme étant l'espérance conditionnelle d'obtenir un certain $\lambda(t)$ étant donné les différents temps d'activité observés

$$\hat{\lambda(t)} = E(\lambda(t)|t_1, t_2, ...) \tag{2.4}$$

où les t_i sont les temps associés aux occurrences du hashtag sur un temps total d'observation que nous noterons par la suite T. De cette équation d'espérance conditionnelle, on peut dériver une équation pour $\hat{\lambda}$ qui prend la forme

$$\begin{aligned} \hat{\lambda}(t) &= f(t) + g(t) \\ &= p(t) \sum_{i:t_i<T} d_i \phi(t - t_i) + d_p p(t) \int_T^t \hat{\lambda}(t)\phi(t - s)ds \end{aligned} \tag{2.5}$$

et devient alors une équation auto-consistante. En examinant cette équation, nous pouvons remarquer que les deux éléments qui la composent décrivent l'activité avant la fin de la période d'observation T et après la fin de cette période d'observation. La première partie

$$f(t) = p(t) \sum_{i:t_i<T} d_i \phi(t - t_i)$$

représente l'intensité du processus sur les données connues décrite par l'équation (2.2) tandis que la seconde partie

$$g(t) = d_p p(t) \int_T^t \hat{\lambda}(t)\phi(t - s)ds$$

décrit le processus d'auto-excitation induit durant la période de prédiction. Dans cette seconde partie, nous pouvons retrouver le caractère auto-consistant de l'équation ainsi

que la constante d_p correspondant à l'espérance conditionnelle du nombre de followers étant donné les temps d'observation inférieurs à T

$$d_p = E(d_i|t_1, t_2, ..)$$

qui peut par exemple être estimé par le nombre moyen de followers sur T. L'équation (2.5) est connue comme équation intégrale linéaire de Volterra du second type. Une méthode de résolution de cette équation auto-consistante sera présentée dans la section 2.3.2.
Nous possédons dès lors tous les ingrédients utiles à la prédiction de l'activité d'un hashtag étant donné l'activité connue jusqu'en un temps T et pouvons alors expliciter cette méthode de prédiction. Étant donnée une largeur fixée pour les intervalles de prédiction après T, notée w, l'activité du hashtag dans le *k-ième* intervalle après T A_k peut être évaluée entre calculant

$$A_k = \int_{T+(k-1)w}^{T+kw} \hat{\lambda}(s)ds, \qquad k = 1, 2, ... \tag{2.6}$$

2.2 Taux d'infection $p(t)$

Comme décrit précédemment, notre modèle considère un paramètre $p(t)$ brièvement décrit ci-avant, appelé le taux d'infection ou encore l'infectiosité du modèle. Ce paramètre est le plus important de notre modèle étant donné que celui-ci va quantifier la viralité du hashtag en un temps donné. Étant donné ce caractère important, nous allons modéliser ce taux d'infection séparément du reste du modèle général et utiliser sa modélisation dans le modèle. Dans un premier temps nous allons présenter la méthode des moyennes mobiles ou moyennes glissantes (moving or running averages) à partir de laquelle nous réaliserons une estimation de l'infectiosité instantanée sur les données. Nous décrirons ensuite le processus de modélisation de l'infectiosité pour terminer par une description de la méthode d'optimisation des paramètres entrant dans la modélisation de celui-ci.

2.2.1 Méthode des moyennes glissantes

Afin d'introduire la méthode des moyennes glissantes, nous allons nous baser sur les sources [15, 16] afin de présenter celle-ci.

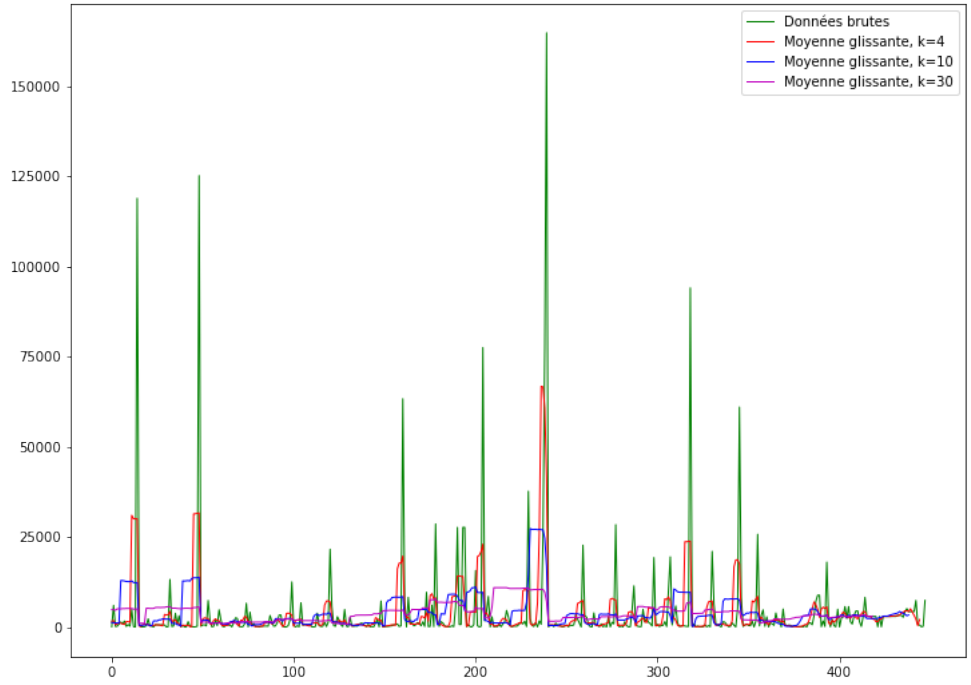


Figure 2.1: Moyennes glissantes pour le nombre de followers des personnes ayant utilisé le hashtag *#sun* évaluées sur 400 heures - échelle linéaire.

Définition 11. *Moyennes glissantes*

Soit une suite de valeurs $\{a_i\}_{i=1,\dots,n}$ et un entier $k \in \mathbb{N}\backslash\{0,1\}$, $k < n$. Les moyennes glissantes associées à l'entier k et à la suite de valeurs $\{a_i\}_{i=1,\dots,n}$ sont une suite de $n-k+1$ éléments s_i calculés en réalisant la moyenne arithmétique d'une sous-suite de k éléments successifs de la suite $\{a_i\}_{i=1,\dots,n}$. Autrement dit, l'élément s_i de la suite des moyennes glissantes est calculé par

$$s_i = \frac{1}{k} \sum_{j=i}^{i+k-1} a_j$$

et est donc la moyenne des k éléments successifs de la suite $\{a_i\}_{i=1,\dots,n}$.

La Figure 2.1 présente la courbe des données du nombre de followers pour les utilisateurs ayant utilisé le hashtag *#sun* ainsi que les moyennes glissantes correspondant aux entiers

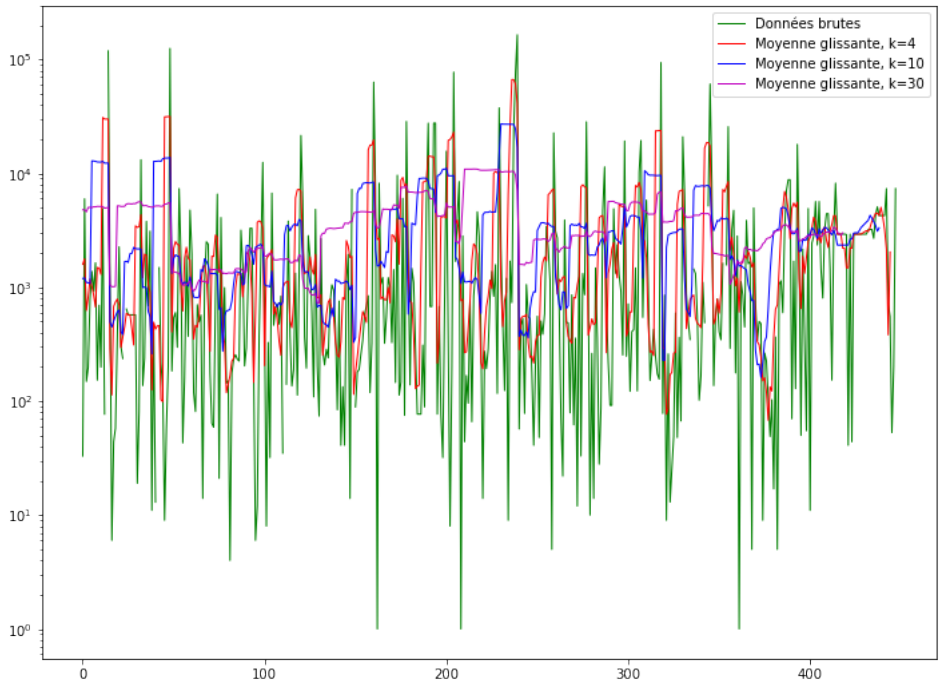


Figure 2.2: Moyennes glissantes pour le nombre de followers des personnes ayant utilisé le hashtag *#sun* évaluées sur 400 heures - échelle logarithmique

$k = 4, 10, 30$. Le code permettant de générer ces résultats est présent dans l'annexe C. Comme nous pouvons le remarquer, la moyenne glissante a un effet lissant sur la courbe par rapport aux données initiales. Cet effet est d'autant plus visible sur la représentation en échelle logarithmique présente à la Figure 2.2.

2.2.2 Estimation du taux d'infection par la méthode des fenêtres glissantes

Afin d'estimer l'infectiosité relative à un hashtag donné, nous allons nous baser sur l'idée sous-jacente de la méthode des moyennes glissantes et mettre celle-ci en application. Tout d'abord, on émet l'hypothèse que le taux d'infection est constant sur de petits intervalles de temps Δt. Considérons que l'on souhaite estimer l'infectiosité de la suite du hashtag (t_i, f_i) où t_i est le temps de réapparition du hashtag et f_i le nombre de followers de la *i-ème* personne réutilisant le hashtag. Afin d'estimer l'infectiosité de la suite du hashtag, on va calculer, pour une taille de fenêtre temporelle donnée, un estimateur dans

une fenêtre temporelle glissante. Contrairement à la méthode des moyennes glissantes, dans laquelle le nombre de valeurs pour calculer la moyenne est systématiquement le même, c'est ici le temps qui va bouger et donc l'estimation se fera sur des nombres différents de valeurs.
L'estimation de l'infectiosité sur un intervalle de temps $[t_{st}, t_{en}]$ de longueur Δt peut être réalisée par la méthode du maximum de vraisemblance en évaluant

$$\hat{p}_{[t_{st},t_{en}]} = \frac{\delta R}{\sum\limits_{i:t_i \in [t_{st},t_{en}]} f_i \left(\Phi(t_{en} - t_i) - \Phi(t_i - t_{st})\right)} \tag{2.7}$$

où δR est le nombre d'occurrences du hashtag entre $[t_{st}, t_{en}]$, t_i le temps associé à la *i-ème* occurrence d'apparition du hashtag et $\Phi(t)$ est l'intégrale du noyau mémoriel défini ci-avant: $\Phi(t) = \int_0^t \phi(s)ds$.
Autrement dit, étant donné la suite du hashtag (t_i, f_i) et une largeur de fenêtre fixée $k \in \mathbb{N}_0$ et le temps total de mesure des réutilisations du hashtag T divisé en intervalles d'une heure à partir de t_0, l'élément p_j de la suite des estimateurs par le maximum de vraisemblance de l'infectiosité sera donné par

$$\hat{p_j} = \frac{\delta R}{\sum\limits_{i:t_i \in [j,j+k]} f_i \left(\Phi((j+k) - t_i) - \Phi(t_i - j)\right)}.$$

Si $T = m$ heures, notre suite d'estimation sera de longueur $m - k + 1$. C'est alors sur cette suite d'estimation et plus particulièrement sur son tracé graphique que nous allons tenter de déceler le comportement de l'infectiosité afin de modéliser la ou les fonctions qui tenteront de coller un maximum aux estimations. Un exemple d'infectiosité générée par la méthode des fenêtres glissantes est disponible à la Figure 2.3.

2.2.3 Modélisation et optimisation des paramètres

Nous allons maintenant nous intéresser, une fois que le modèle a été choisi, à la manière d'en déterminer les paramètres idéaux. En effet, de nombreux comportements peuvent se retrouver dans l'infectiosité: comportements oscillatoires, comportements constants, comportements en loi de puissance, décroissance exponentielle mais les paramètres cachés derrière ces comportements nécessitent un ajustement. Par exemple, si on choisit de modéliser le comportement oscillatoire décelé dans les estimations par un sinus, l'amplitude de celui-ci devra être ajustée, ou si on choisit d'exprimer la disparition progressive d'un

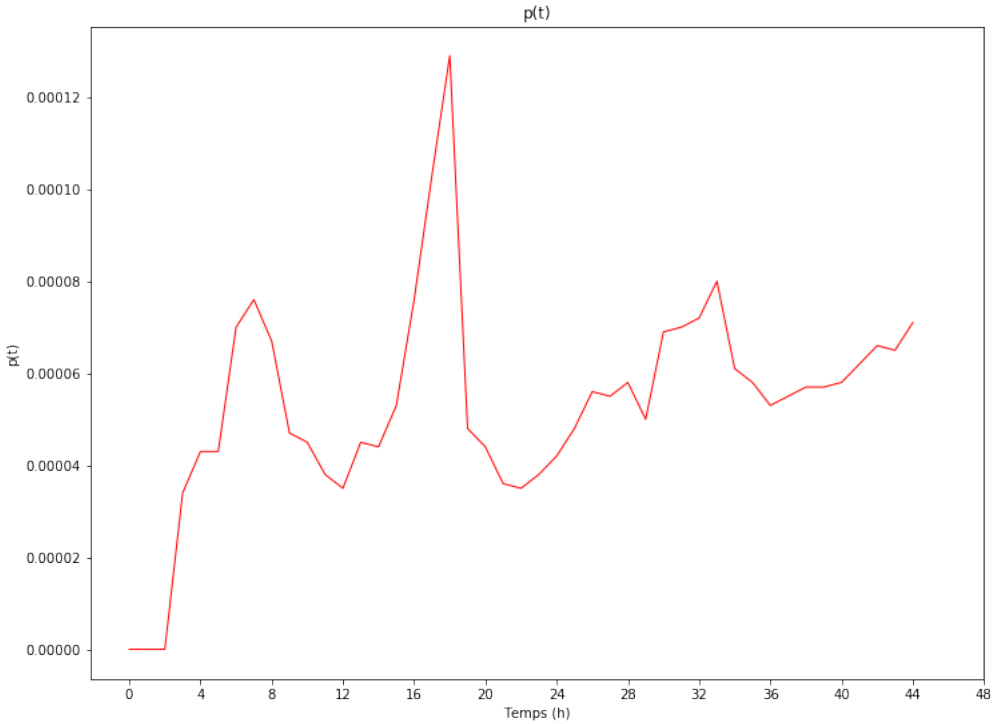


Figure 2.3: Infectiosité estimée par la méthode des fenêtres glissantes sur le hashtag $\#BEL$ sur une période de 48h

hashtag sur le réseau par une exponentielle décroissante, un paramètre représentant l'importance de cette décroissance devra être ajusté.

Considérons

$$p(t; r_1, r_2, ..., r_l)$$

le modèle d'infectiosité à l paramètres réalisé sur base de la suite d'estimation de l'infectiosité. Une méthode permettant d'ajuster les l paramètres du modèle à l'infectiosité estimée produite par les données empiriques est de minimiser l'erreur des moindres carrés entre la fonction $p(t; r_1, r_2, ..., r_l)$ et la suite des estimations $(\hat{p}_i)$ donnée par

$$E_p = \sum_{j=0}^{M-1} \left(\hat{p}_j - p((j+0.5)k; r_1, r_2, ..., r_l)\right)^2 \tag{2.8}$$

où $M = \dfrac{m}{k}$ est le nombre de fenêtres, $\hat{p}_j$ est l'estimation du taux d'infection dans

$[jk, (j+1)k]$ et où $(j+0.5)k$ représente le centre de la *j-ème* fenêtre.
La minimisation de l'équation (2.8) par rapport aux paramètres $(r_1, r_2, ..., r_l)$ peut être réalisée de manière numérique à l'aide de plusieurs méthodes comme par exemple les régions de confiance ou les descentes de gradient. Il existe cependant une famille de problèmes nommés ajustement de courbes au sens des moindres carrés ("least squares curve fitting" en anglais) qui possède des méthodes de résolution adaptées comme par exemple l'algorithme de Levenberg-Marquardt décrit dans la section 2.3.3. C'est cet algorithme qui sera utilisé afin de trouver la valeur optimale des paramètres de nos modèles.

2.3 Outils numériques

2.3.1 Noyau mémoriel: forme, estimation et paramètres

Le nom du noyau mémoriel provient de l'objectif poursuivi par celui-ci. Le noyau mémoriel est une mesure qui tente de représenter la mémoire d'un système physique ou social à un stimuli [17]. Dans le cas des réseaux sociaux, le noyau mémoriel va chercher à modéliser le laps de temps pouvant exister entre le moment où un élément apparaît dans le fil d'actualité d'un utilisateur et le moment où l'utilisateur choisit de faire suivre l'élément en question. On peut donc dire que le noyau mémoriel modélise le temps de réaction d'un humain étant exposé à des contenus sur les réseaux sociaux. Ce noyau mémoriel représente donc une probabilité, qui est la probabilité pour un temps s donné que l'utilisateur ayant été exposé à un certain contenu réutilise ce contenu à son tour en ce temps s. Le noyau mémoriel est, comme évoqué ci-précédemment, noté $\phi(s)$.
Dans le cadre des réseaux sociaux, on considère que le temps entre l'arrivée d'un contenu en $t = 0$ dans le fil d'actualité d'un individu et son partage au temps s par l'utilisateur suit une densité de probabilité $\phi(s)$. Il a été démontré que cette probabilité suivait, pour les réseaux sociaux, une loi dite de "queue lourde"[18] (heavy-tailed distribution en anglais). De plus, le noyau mémoriel est propre à chaque réseau et celui-ci possède une forme unique propre à chaque réseau.[13] Pour le réseau social Twitter, duquel proviennent nos données, le noyau mémoriel présente une forme de queue lourde précédée par environ cinq minutes où celui-ci est constant. Tous ces éléments nous permettent d'obtenir une forme définie pour le noyau mémoriel associé à Twitter, constante pendant

un temps bien défini puis décroissante en queue lourde. Sa forme est donnée par

$$\phi(s) = \begin{cases} 0 & \text{si } s < 0 \\ c_0 & \text{si } 0 \leq s \leq s_0 \\ c_0 \left(\frac{s}{s_0}\right)^{-(1+\theta)} & \text{sinon} \end{cases} \tag{2.9}$$

Les paramètres (s_0, c_0, θ) sont estimés par des données empiriques du réseau et valent, pour Twitter, $(s_0, c_0, \theta) = (0.0833, 6.49 \times 10^{-4}, 0.242)$ [14].

2.3.2 Résolution de l'équation intégrale linéaire de Volterra du second type

Présentons maintenant la méthode utilisée pour résoudre l'équation auto-consistante (2.5) permettant d'estimer l'intensité du processus. Comme énoncé ci-avant, la forme de cette équation correspond à la forme d'une équation intégrale linéaire de Volterra du second type et la méthode présente dans [19] sur laquelle nous nous basons propose une méthode de résolution basée sur la méthode de Newton-Cotes d'ordre 1.

Soit l'équation intégrale de Volterra du second ordre

$$f(t) = \int_a^t K(s,t) f(s) ds + g(t) \tag{2.10}$$

pour laquelle la fonction f est inconnue et la fonction K est connue. Considérons la grille à espacement uniforme définie par

$$t_i = a + ih, \quad i = 1, ..., N,$$

$$h = \frac{t-a}{N}$$

où $N > 0$ est le nombre de subdivisions souhaitées de l'intervalle $[t, a]$ pour intégrer.
Pour la grille considérée, choisissons comme quadrature la méthode de Newton-Cotes d'ordre 1 plus connue sous le nom de méthode des trapèzes, qui interpole la fonction à intégrer par un polynôme de degré 1 sur chaque intervalle, formant ainsi une suite de trapèzes (Figure 2.4). L'intégration d'une fonction f entre a et t par la formule composite des trapèzes est donnée par

$$\int_{a=x_1}^{t=x_N} f(x) dx = h \left(\frac{1}{2} f_1 + f_2 + ... + f_{N-1} + \frac{1}{2} f_N \right) \tag{2.11}$$

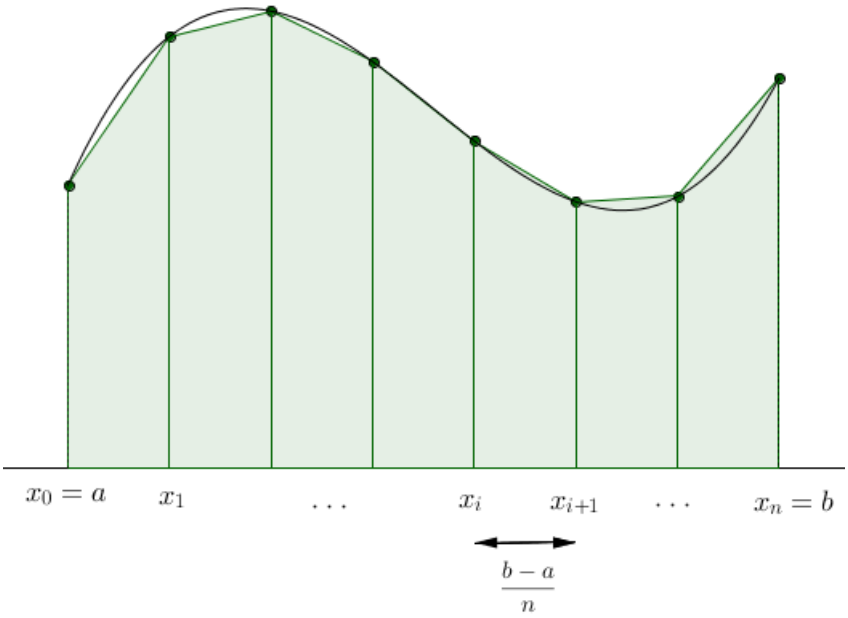


Figure 2.4: Illustation de l'intégration par la méthode des trapèzes. Source:[20]

où $f_i = f(x_i)$. L'équation de l'intégrale présente dans (2.10) peut alors être décomposée par la formule des trapèzes et donne

$$\int_a^{t_i} K(t_i, s)f(s)ds = h\left(\frac{1}{2}K_{i0}f_0 + \frac{1}{2}K_{ii} + \sum_{j=1}^{i-1} K_{ij}f_j\right). \tag{2.12}$$

Cette formulation injectée dans l'équation (2.5) permet d'obtenir l'équation itérative d'évaluation de f, donnée par

$$\begin{cases} f_0 = g_0 \\ f_i = \left(g_i + h\left(\frac{1}{2}K_{i0}f_0 + \sum_{j=1}^{i-1} K_{ij}f_j\right)\right)\left(1 - \frac{1}{2}hK_{ii}\right)^{-1} \end{cases} \tag{2.13}$$

et permet par conséquent d'évaluer numériquement la valeur de la fonction f.

2.3.3 Ajustement de courbes et méthode de Levenberg-Marquardt

Nous allons désormais présenter la théorie de l'ajustement de courbes et plus précisément l'ajustement non-linéaire au sens des moindres carrés et décrire une méthode

d'ajustement de courbes appelée méthode de Levenberg-Marquardt, développée par Kenneth Levenberg (1944) et ensuite par Donald Marquardt (1963) [21].

Soit un ensemble de m points (x_i, y_i) appelé ensemble de données ou encore données et soit $\hat{y}(x_i, p)$ une fonction non-linéaire à n paramètres $p_1, ..., p_n$ à ajuster sur les données. L'ajustement de la courbe $\hat{y}$ au sens des moindres carrés consiste à déterminer la valeur des n paramètres $p_1, ..., p_n$ de telle sorte que la somme des carrés des résidus $(r_i)_{i=1}^m$ soit minimale. Le *i-ème* résidu est défini comme la différence entre la valeur effective de l'observation y_i et la valeur de la fonction à ajuster pour le point de x correspondant, i.e.

$$r_i = y_i - \hat{y}(x_i, p).$$

Autrement dit, un problème d'ajustement de courbe au sens des moindres carrés revient à un problème d'optimisation (minimisation) de l'erreur des moindres carrés entre les données y_i et la fonction cherchant à les approximer par rapport aux paramètres de celle-ci. Nous connaissons déjà plusieurs méthodes itératives de minimisation, comme par exemple la méthode du gradient ou la méthode de Gauss-Newton [22]. Dans la méthode du gradient, les paramètres seront optimisés en suivant la direction de la plus forte pente tandis que dans la méthode de Gauss-Newton, la somme des carrés est minimisée en considérant que la fonction est localement quadratique autour du minimum et en minimisant la forme quadratique de la fonction considérée. [22, 23]
L'idée de Levenberg, suivi par Marquardt, était de jumeler la méthode du gradient avec la méthode de Gauss-Newton afin d'améliorer les performances de convergence de l'algorithme. Rappelons tout d'abord comment sont définies la mise à jour vers la solution dans le cas des deux méthodes composant la méthode de Levenberg-Marquardt. Dans le cas de la descente de gradient, la mise à jour vers la solution est donnée par

$$h_{gd} = \alpha J^T (y - \hat{y})$$

où J est la matrice jacobienne de $\hat{y}$ par rapport aux paramètres p et où α est la longueur de la descente le long de la plus forte pente. Dans le cas de la méthode de Gauss-Newton, la mise à jour est donnée en résolvant

$$(J^T J) h_{gn} = J^T (y - \hat{y}).$$

Dans le cas de Levenberg-Marquardt, la mise à jour proposée est donnée par

$$(J^T J + \lambda J^T J) h_{lm} = J^T (y - \hat{y}). \tag{2.14}$$

Dans cette formule de mise à jour, nous remarquons l'apparition d'un nouveau paramètre, λ, appelé paramètre d'amortissement de Levenberg-Marquardt ("damping parameter" en anglais) et c'est ce paramètre qui va déterminer si la méthode utilisée pour la mise à jour en cours est plutôt de type descente de gradient ou plutôt de type Gauss-Newton. La méthode de Levenberg-Marquardt représente donc une combinaison linéaire des deux méthodes. Lorsque le paramètre d'amortissement λ est grand, la mise à jour se fait grâce à $h_{lm} \simeq -\frac{1}{\lambda} J^T(y - \hat{y})$ i.e on avance légèrement dans la direction de la plus forte pente. Dans ce cas de figure, nous nous trouvons dans le cas où la descente de gradient prend le dessus sur la méthode de Gauss-Newton. A contrario, lorsque λ est très petit, la mise à jour s'apparente à une mise à jour selon Gauss-Newton, i.e. $h_{lm} \simeq h_{gn}$ et dans ce cas c'est la méthode de Gauss-Newton qui prend le dessus sur la méthode de descente de gradient. C'est en quelque sorte les étapes de descente de gradient qui dominent lorsque l'on est éloigné de la solution jusqu'à ce que l'itération en cours soit suffisamment proche de la solution, après quoi un changement de comportement s'effectue et c'est alors la méthode de Gauss-Newton qui prend le pas afin de se rapprocher de la solution et de terminer l'approximation. [24]

2.4 Synthèse

Nous allons maintenant résumer la philosophie du modèle que nous avons présenté en décrivant celui-ci point par point. Pour un hashtag donné,

1. à partir des données collectées, le taux d'infection est estimé par la méthode des fenêtres glissantes par l'estimateur du maximum de vraisemblance,
2. l'infectiosité estimée est tracée et sur base de la forme de cette représentation graphique, un modèle d'infectiosité $p(t)$ est proposé,
3. les paramètres du modèle d'infectiosité sont ajustés en minimisant l'erreur des moindres carrés entre la courbe proposée et les données empiriques, par exemple en utilisant la méthode de Levenberg-Marquardt,
4. l'intensité du processus est évaluée en résolvant l'équation auto-consistante (2.5),
5. l'activité future est évaluée en intégrant l'intensité du processus sur l'intervalle de temps considéré.

Afin d'obtenir l'activité future sur un intervalle de temps T_{pred} débutant en T et pour une longueur d'intervalles de prédiction w fixée, on va donc successivement calculer les valeurs de l'intégrale

$$A_k = \int_{T+(k)w}^{T+(k+1)w} \hat{\lambda}(s)ds, \qquad k = 0, 1, 2, ..., \left\lceil \frac{T_{pred}}{w} \right\rceil \tag{2.15}$$

où $\left\lceil \frac{T_{pred}}{w} \right\rceil$ est la notation associée à la partie entière supérieure par excès de $\frac{T_{pred}}{w}$.

Chapitre 3

Données et traitement

Dans ce chapitre, nous allons présenter en détail les méthodes utilisées afin de collecter les données utilisées et présenter la manière dont celles-ci ont été pré-traitées afin d'être utilisées pour la modélisation et le calibrage de l'infectiosité. Nous réaliserons également une première étude sur les données récoltées afin de présenter les dynamiques qui peuvent exister dans les hashtags.

3.1 L'API Twitter et le module Python Tweepy

Dans un premier temps, nous allons brièvement définir ce qu'est une API et expliquer en quoi l'API Twitter va nous servir pour construire notre base de données. Dans le monde informatique, API est l'acronyme de "application programming interface" traduit en français par interface de programmation applicative. Sur base de [25, 26, 27] nous pouvons proposer la définition suivante d'API:

Définition 12. *Interface de programmation applicative (API)*

Une interface de programmation applicative est un ensemble de processus, de classes, de méthodes, de fonctions, de sous-routines à partir desquelles deux logiciels interagissent. Une API permet à des programmes consommateurs d'utiliser ces éléments afin d'accéder à des programmes fournisseurs.

L'API peut donc être vue comme un produit. C'est une interface qui une fois fournie à l'utilisateur lui permet d'obtenir des résultats par simple appel de fonction ou de routine.

Elle lui permet donc d'accéder à des services implémentés en back-end.
Selon [25], il existe quatre catégories d'APIs: API de détection, d'enrichissement, de perception et d'action. L'API de Twitter qui nous intéresse, appelée Search API, est une API d'enrichissement. En effet, celle-ci nous permet d'enrichir et d'élargir les données pour obtenir une meilleure compréhension des faits en cours. La Search API de Twitter [28] permet de récupérer des données historiques (les tweets) sur des données remontant jusqu'à 30 jours pour les versions gratuites de celle-ci. Nous allons maintenant détailler brièvement le fonctionnement de cette API et décrire le module Python que nous avons utilisé afin de récupérer les données, nommé Tweepy.

L'API Twitter Search est une query-based API, ce qui signifie que les routines cachées derrière les fonctionnalités offertes par celle-ci cherchent à répondre à des requêtes effectuées envers la base de données contenant les tweets. Notons que pour de larges requêtes - par exemple un hashtag particulièrement populaire - les requêtes ne retournent que très rarement la totalité des tweets correspondant. L'API réalise donc parfois des opérations de sampling i.e. elle retourne un échantillon et pas un ensemble complet de tweet[28, 29]. Les requêtes peuvent contenir plusieurs restrictions comme l'intervalle de temps sur lequel on recherche les tweets, la zone géographique considérée, la langue d'écriture et bien plus encore [30].
Afin de faciliter l'utilisation de cette API et de pouvoir en retirer le meilleur, l'utilisation d'un module Python nommé Tweepy a été préférée. Ce module permet de réaliser la connexion d'un utilisateur vers l'API et de faciliter la communication avec celle-ci afin de réaliser les requêtes. L'annexe A présente la routine créée sur base des fonctions du module Tweepy afin de récupérer les tweets d'un hashtag donné et de créer le fichier de récolte de données.

3.2 Pré-traitement des données

Une fois les données récoltées par le code présenté dans l'annexe A, celles-ci doivent être mises au format que requiert l'exécution des différents programmes. Comme une partie de notre travail sera réalisé en utilisant le code existant du modèle de [14], nous avons décidé de calquer le type de fichier d'entrée de la suite de nos processus sur celui-ci. Un template de fichier de récolte de données peut être visualisé dans la Table 3.1 et le format d'utilisation à obtenir après pré-traitement de données est disponible dans la Table 3.2. Dans la Table 3.1, les données sont présentes telles qu'elles ont été récoltées sur Twitter. La colonne *timestamp* contient les temps auxquels les tweets contenant

$timestamp$	$followers$
$time_1$	$n\_followers_1$
$time_2$	$n\_followers_2$
$time_3$	$n\_followers_3$
$\vdots$	$\vdots$
$time_n$	$n\_followers_n$

Table 3.1: Template de données récupérées de Twitter

total reuse of hashtag	hour of first measured use
$t_0 = 0$	n_0
t_1	n_1
t_2	n_2
$\vdots$	$\vdots$
t_T	n_T

Table 3.2: Template de données récupérées de Twitter, $t_0 \leq t_1 \leq ... \leq t_T$

le hashtag considéré ont été postés. Ces temps ont été convertis en Unix timestamps i.e. ils représentent le nombre de secondes écoulées depuis le 01/01/1970 00:00:00. La seconde colonne associe quant à elle le nombre de followers de l'utilisateur ayant utilisé le hashtag au temps de la ligne correspondante.

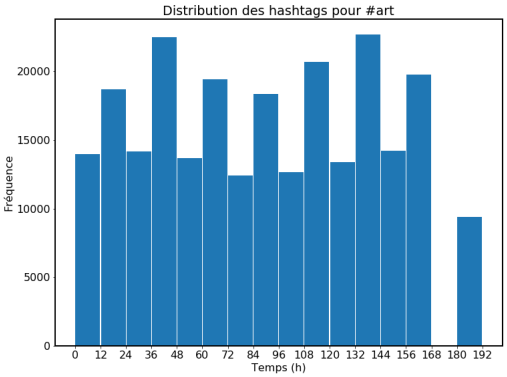
Afin d'obtenir un fichier utilisable par le code proposé dans [14], il était nécessaire de convertir le fichier obtenu vers un fichier dont le format était du type représenté à la Table 3.2. Pour ce faire, le module Python Pandas a été utilisé. Ce module permet entre autres de réaliser des importations, créations, exportations de dataframes et de travailler sur les données de ceux-ci. Dans un premier temps, les données initiales ont simplement été classées par ordre croissant selon la première colonne. Ensuite, afin que tous les temps apparaissent en tant que temps relatifs au premier temps mesuré, la valeur du timestamp de la première mesure a été retirée à toutes les valeurs. Les données temporelles ont également subi une conversion afin que le nombre de secondes séparant une utilisation du hashtag de la première utilisation mesurée soit mesuré en heures. Finalement, la première ligne, contenant le nombre total de réutilisation du hashtag sur l'intervalle de temps considéré ainsi que l'heure à laquelle celui-ci a été posté sont ajoutées en première ligne. L'annexe B présente les sous-routines et la routine Python réalisées afin de parfaire ces objectifs de prétraitement.

3.3 Présentation des données

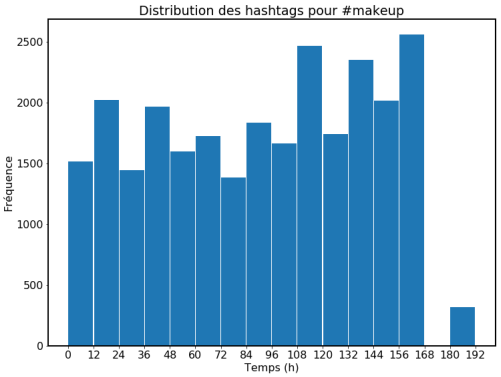
Dans cette partie de notre travail, nous allons décliner l'ensemble de hashtags qui nous a intéressé. Afin d'obtenir un large éventail de possibilités de comportements, notre collecte de données s'est concentrée sur plusieurs aspects du réseau social Twitter. La quasi totalité des hashtags récoltés ont été prospectés entre la date du 12 juillet 2018 00:00:00 et le 19 juillet 2018 23:59:59. Sauf contre indication future, ces dates seront les dates de collectes considérées par la suite. Nous nous sommes également limités à collecter des hashtags écrits uniquement en anglais, et ce pour éviter de jongler avec de trop grandes bases de données susceptibles de poser des problèmes d'exécution des différents scripts par la suite, mais aussi car les utilisateurs de Twitter sont en grande partie anglophones [31].

Dans un premier temps, nous avons tenté de récolter des hashtags correspondant à des événements ponctuels. Par exemple, les hashtags *#REDTOGETHER* ou *#BELENG* qui sont tous deux liés à l'événement de la Coupe du monde de la FIFA 2018, mais aussi *#WorldEmojiDay* faisant référence à la journée mondiale des Emojis, s'étant déroulée le mardi 17 juillet 2018. Un autre exemple est le *#trump*, faisant référence à D. Trump, qui a été largement utilisé suite aux différents événements s'étant déroulés dans cette période (sommet de l'OTAN, discussions avec le président russe V. Putin,...).
Ensuite, nous nous sommes concentrés sur un panel de hashtags dit "les plus populaires". Afin de réaliser une liste d'un sous-ensemble de hashtags populaires, nous nous sommes basés sur la liste des hashtags populaires du réseau social Instagram [32] ainsi que sur les hashtags populaires présentés par la page Twitter [33] présentant les top hashtags de France, Allemagne, Espagne, Australie et Pologne. Une fois cette sélection réalisée, nous obtenons la liste exhaustive de 91 hashtags que nous avons utilisés, disponible ci-après.

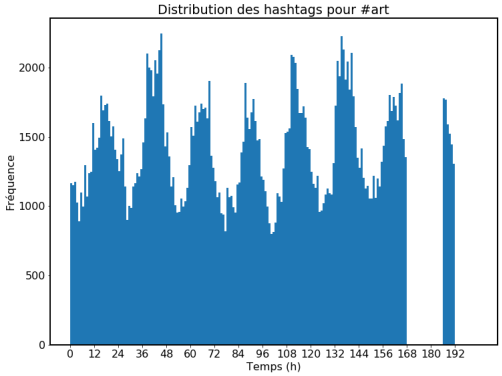
Liste finale des hashtags: *#amazing, #art, #ausvotes, #aviation, #baby, #beach, #beauty, #bestoftheday, #birthday, #black, #blackandwhite, #blue, #cat, #ChoiceInternationalArtist, #christmas, #classical, #cleaneating, #cm2018, #cocktails, #competition, #contest, #contests, #cool, #cute, #design, #dog, #fit, #fitness, #flowers, #food, #foodie, #foodporn, #friends, #fun, #funny, #girl, #girls, #guitar, #gym, #happy, #happyhour, #healthy, #holidays, #hot, #hungry, #LebronJames, #life, #lol, #love, #makeup, #marketing, #motivation, #music, #musician, #myhealthrecord, #night, #nofilter, #party, #pasta, #photo, #photooftheday, #picoftheday, #pink, #pizza, #pretty, #promotion, #red, #REDTOGETHER, #selfie, #shoppingaddict, #singer, #singing, #sky, #smile, #sun, #sunset, #sushi, #swag, #sweet, #talentedmusicians,*



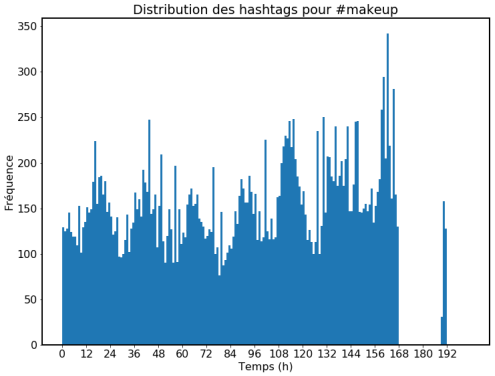
(a) Histogramme de *#art* par intervalles de 12h



(b) Histogramme de *#makeup* par intervalles de 12h



(c) Histogramme de *#art* par intervalles de 1h



(d) Histogramme de *#makeup* par intervalles de 1h

Figure 3.1: Histogramme de données défectueuses

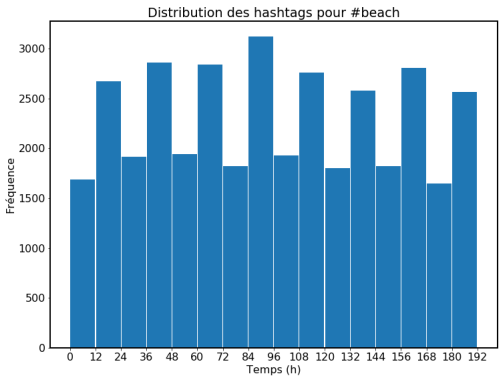
#Tomorrowland, #travel, #trump, #vegetarian, #wedding, #win, #wine, #work, #workout, #WorldEmojiDay, #yummy.

Une fois la collecte effectuée pour ces différents hashtags, nous avons cherché à déceler les différentes dynamiques qu'il était possible de retrouver dans la distribution des hashtags, pour ce faire, nous avons affiché deux histogrammes de données pour chaque hashtag, le premier en considérant des largeurs de bâtonnets fixées à 12 heures et le second en fixant les tailles de bâtonnets à 1 heure. Cet affichage de données nous a permis de faire plusieurs constatations. La première constatation est que malheureusement, malgré la bonne exécution du script de récupération des données, certains d'entre eux ont arrêté de collecter des données pendant un court intervalle de temps (Figure 3.1). Fort heureusement, ce phénomène n'a touché que très peu de nos récoltes de donnés (moins de 5%) ce qui nous laisse une base de données de grande envergure pour réaliser la suite de notre analyse. Nous tenterons cependant d'utiliser ces données afin d'observer si notre modèle tend à produire des résultats cohérents avec les autres dynamiques observées.

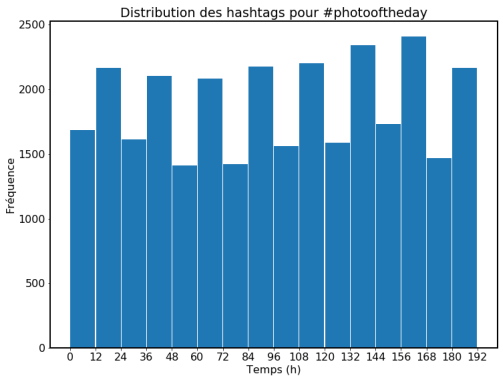
Nous avons ensuite tenté de classer les hashtags restants en fonction de la forme de ceux-ci. Nous avons choisi de considérer deux types de dynamiques différentes. Dans un premier temps, nous avons pu remarquer une dynamique fort présente dans les hashtags récoltés, de type oscillatoire sur 24 heures présentant l'importance des cycles à alternance diurne-nocturne, aussi appelés cycles circadiens. La Figure 3.2 présente une sélection de dynamiques oscillatoires en histogramme de 12 heures et la Figure 3.3 présente une sélection de dynamiques oscillatoires en histogramme de 1 heure. Nous constatons que ces dynamiques sont principalement liées aux hashtags dits "all time popular" (tout le temps populaires) i.e. des hashtags qui peuvent être reliés à des événements ponctuels comme ils peuvent également ne pas l'être. Ce sont en quelque sorte les hashtags "de base" du réseau, présents en tout temps.

3.3.1 Dynamiques particulières

Au-delà des dynamiques oscillatoires sur 24 heures des "all time popular hashtags", nous avons pu déceler des comportements un peu plus singuliers au sein des représentations graphiques que nous avons obtenues. En plus de présenter les différentes figures correspondant aux comportements remarqués, nous allons chercher à expliquer le pourquoi du comportement remarqué. Dans un soucis de lisibilité, nous ne présenterons ici que les histogrammes par intervalles de 12 heures.

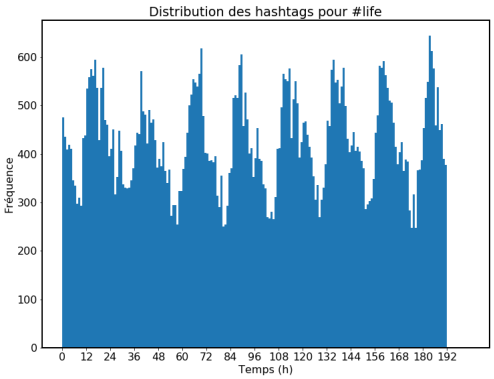


(a) Histogramme de *#beach* par intervalles de 12h

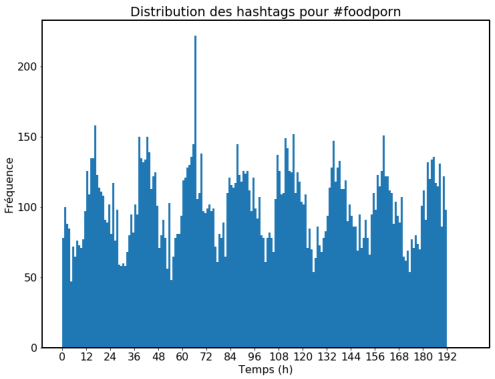


(b) Histogramme de *#photooftheday* par intervalles de 12h

Figure 3.2: Histogrammes de hashtags à dynamique oscillatoire sur 24h



(a) Histogramme de *#life* par intervalles de 1h



(b) Histogramme de *#foodporn* par intervalles de 1h

Figure 3.3: Histogramme de hashtags à dynamique oscillatoire sur 24h

Tout d'abord, dans la Figure 3.4a pour le hashtag *#REDTOGETHER*, qui est relié à la Coupe du monde de football 2018, nous remarquons une activité particulièrement forte entre 48 et 60 heures et entre 72 et 84 heures. Ces heures correspondent à la veille du match Belgique-Angleterre et au jour du match Belgique-Angleterre respectivement. L'activité de ce hashtag a donc été accrue au moment de l'événement sportif auquel celui-ci est lié, ce qui est un comportement tout à fait cohérent. Ensuite, la Figure 3.4c qui représente le hashtag *#cm2018* possède elle aussi des pics de concentration mais ceux-ci sont décalés de 24 heures par rapport à ceux concernant la Belgique (les pics se trouvent entre 72 et 84 heures puis entre 96 et 108 heures), ce qui correspond avec la veille de la Coupe du monde et au soir du match de finale de Coupe du monde pour les français. Dans la suite des événements sportifs, nous pouvons voir à la Figure 3.4b la dynamique du hashtag *#LebronJames*, célèbre basketteur américain évoluant actuellement chez les Los Angeles Lakers. Ce hashtag a été capturé entre le 30 juillet 2018 et le 6 août 2018 suite aux événements de polémique liés au basketteur et au président américain D. Trump qui se sont déroulés le 4 août et dans lesquels les deux célébrités se sont échangés plusieurs piques sur le réseau social. Cet échange ainsi que la prise de position de la femme du président M. Trump pour défendre le basketteur ont eu un effet viral sur le réseau, marqué par une hausse significative d'activité entre 108 et 156 heures.

Ensuite, la Figure 3.5b et la Figure 3.4d nous montrent une dynamique bien particulière. Ces deux figures, correspondant aux hashtags *#cocktails* et *#happyhour*, présentent une hausse de popularité des deux hashtag sur le jeudi 12 et vendredi 13 juillet, une perte de popularité sur le samedi et le dimanche puis finalement un regain de popularité pour le début de la semaine suivante. Cette dynamique peut être expliquée par des aspects de marketing. En effet, la stratégie d'happy hour a pour objectif d'attirer les consommateurs vers les endroits de fête (bars, pubs, boîtes de nuits,...) les jours où la fréquentation de ce type d'établissements est moindre, typiquement en semaine lorsque les gens travaillent. Il s'agit d'attirer plus de clients avec des promotions e.g. une consommation achetée une offerte, une consommation achetée la deuxième à moitié prix, etc. Ce genre de stratégies commerciales ne sont pas nécessaires les "jours de fête" i.e. les week-ends. C'est cet aspect marketing qui permet d'expliquer ce creux d'utilisation du hashtag pendant les heures correspondant à des week-ends.

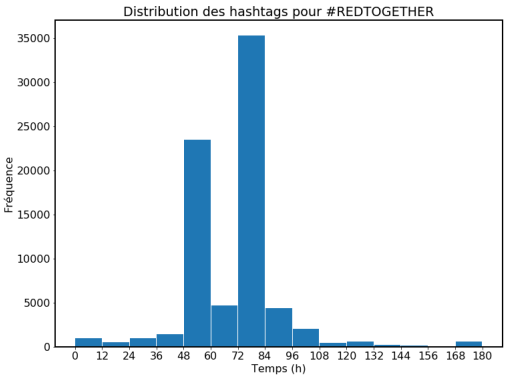
Le *#myhealthrecord* représenté à la Figure 3.5a décrit l'activité du nouveau hashtag associé à la base de données australienne liée aux soins de santé. Selon [34], My Health

Record est une nouvelle base de données gouvernementale australienne dont le but est de stocker les données historiques de soins de santé associées à la population australienne. L'idée est de permettre aux médecins et au personnel médical d'avoir accès direct au dossier médical d'un individu dans un cadre de soins bien précis. L'un des objectifs de ce système est de réduire le nombre d'erreurs médicales comme par exemple en réduisant les risques de mauvaises prescriptions.

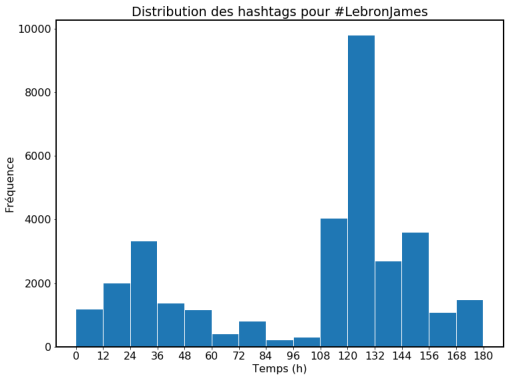
Pourquoi cette explosion soudaine du nombre de hashtags ? La mise en place de My Health Record a été voté à l'unanimité par le gouvernement australien en août 2017 [35]. Il est donc légitime de s'interroger sur le pourquoi de cette explosion du hashtag à partir du 15 juillet 2018. L'explication est simple: tous les australiens ne sont pas en accord avec la mise en place de cette base de données généralisée. Plusieurs raisons motivent leur désaccord, comme par exemple un aspect de sécurité et d'accessibilité à cette base de données, mais également car certains citoyens ne souhaitent pas que leurs informations soient accessibles à tous. L'explosion du hashtag s'explique par le fait que c'est à partir du 16 juillet 2018 et pendant une période de trois mois que les citoyens australiens pourront refuser d'être inscrits dans cette base de données. Passé ce délai, tous les citoyens n'y étant pas encore inscrits et n'ayant pas manifesté le souhait de ne pas y figurer y seront automatiquement ajoutés. L'effet de cette explosion sur le réseau est donc provoqué par l'ouverture de la période de refus d'inscription dans cette base de données.

La Figure 3.5c liée au hashtag *#ChoiceInternationalArtist* des Teen Choice Awards présente elle aussi une dynamique particulière de décroissance avec de légers regains d'utilisation ça et là. Nous n'avons malheureusement pas réussi avec les actualités de la période considérée à expliquer cette tendance. Finalement, la Figure 3.5d présente les hashtags liés au festival de musique électronique Tomorrowland. Malgré que nous n'arrivions pas à expliquer l'entièreté de la dynamique observée sur les huit jours considérés, nous pouvons remarquer de forts pics d'utilisation du hashtag vers la fin de la période d'observation (18 et 19 juillet) qui correspondent à la veille et au jour de début de l'événement.

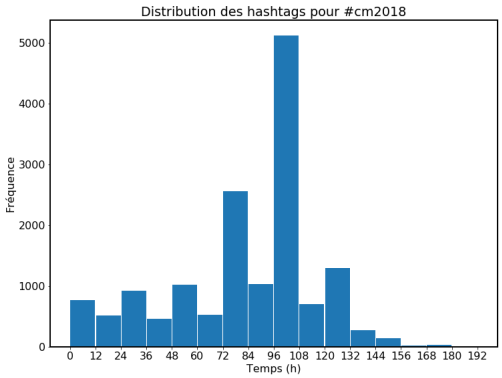
Au terme de cette section, nous remarquons finalement que dans notre panel de hashtags, nous possédons quatre types de dynamiques différentes: les hashtags "all time popular", les hashtags présentant un défaut de récolte de données, les hashtags à dynamique particulière liés à un événement ponctuel (événement sportif, concert, etc.) et les hashtags à dynamique hebdomadaire.



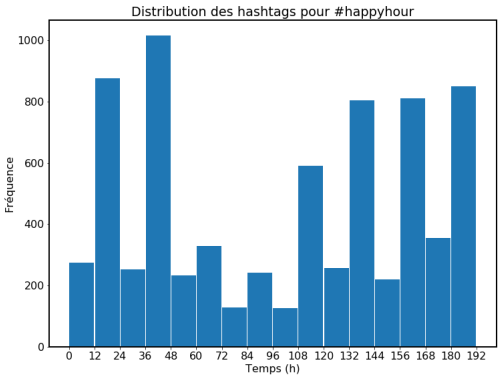
(a) Histogramme de *#REDTOGETHER* par intervalles de 12h



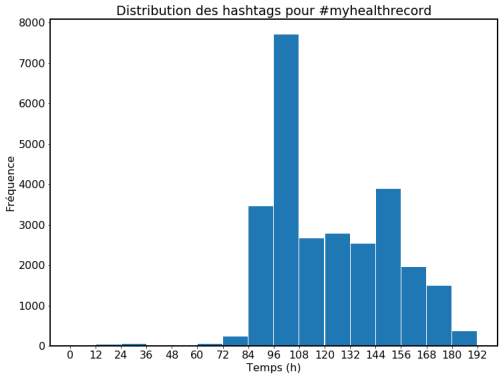
(b) Histogramme de *#LebronJames* par intervalles de 12h



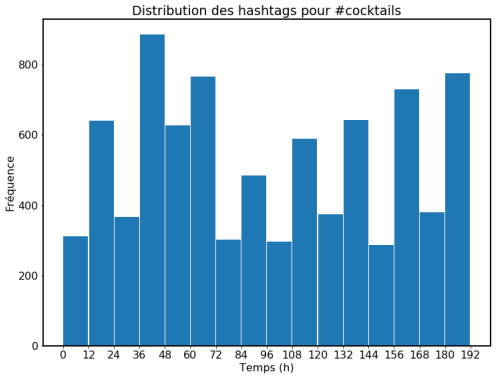
(c) Histogramme de *#cm2018* par intervalles de 12h



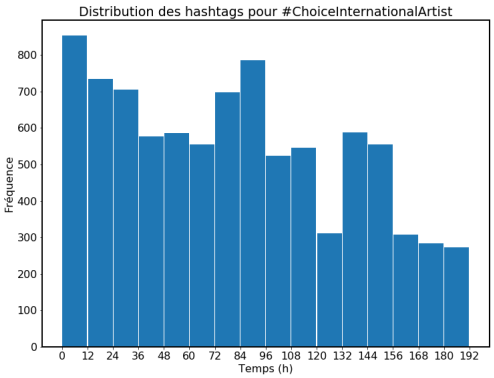
(d) Histogramme de *#happyhour* par intervalles de 12h



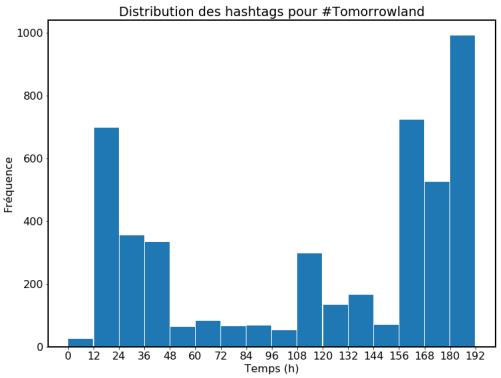
(a) Histogramme de *#myhealthrecord* par intervalles de 12h



(b) Histogramme de *#cocktails* par intervalles de 12h



(c) Histogramme de *#ChoiceInternational-Artist* par intervalles de 12h



(d) Histogramme de *#Tomorrowland* par intervalles de 12h

Figure 3.5: Comportements de données singuliers 2

Chapitre 4

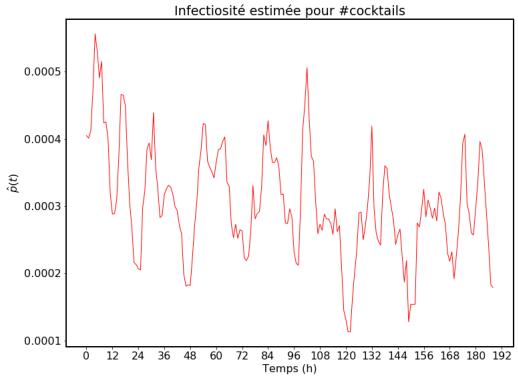
Modélisation de l'infectiosité

Les objectifs de ce chapitre sont multiples mais cependant tous étroitement liés. Notre but au terme de ce chapitre est de proposer une méthode automatisée de modélisation de l'infectiosité qui sera valide quel que soit le comportement de celle-ci. Pour ce faire, nous utiliserons l'estimateur du maximum de vraisemblance qui a été défini dans la section 2.2.2. Nous utiliserons également des outils appartenant à l'analyse complexe, au machine learning ainsi qu'au monde de l'analyse numérique. L'un de ces outils a, au sein de la section 2.3.3, déjà été présenté. Il s'agit de l'algorithme de Levenberg-Marquardt utilisé dans le cadre de l'ajustement de courbes. Les autres outils utilisés seront introduits en temps utile, le moment où ceux-ci ont été introduits étant judicieux pour comprendre le processus qui a été mis en place pour obtenir une modélisation cohérente de l'infectiosité. Ce chapitre sera dès lors décliné en 3 parties qui concerneront successivement l'analyse des taux d'infectiosité liés aux données récoltées et une critique des résultats obtenus, une modélisation de celui-ci basée sur la théorie des séries de Fourier et ensuite, étant donné les résultats obtenus en utilisant les séries de Fourier pour la modélisation, nous critiquerons ceux-ci et introduirons les concepts d'overfitting et d'underfitting liés au monde du machine learning ce qui nous conduira à obtenir une méthode automatisée de modélisation du taux d'infection. L'enchaînement de ces méthodologies a pour but d'améliorer successivement l'automatisation et la qualité de la modélisation.

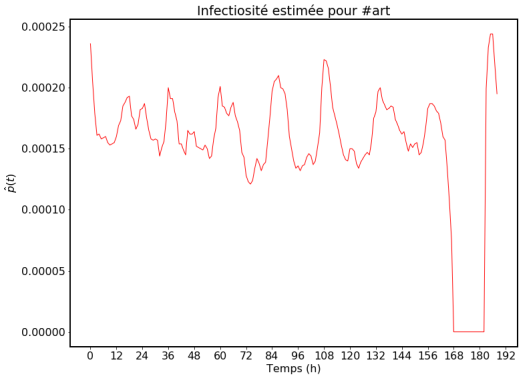
4.1 Estimateurs du maximum de vraisemblance par les fenêtres glissantes

Dans cette première partie, nous allons présenter une sélection d'infectiosités liées aux hashtags listés dans la section 3.3. Les estimations de l'infectiosité ont été évaluées en utilisant l'estimateur du maximum de vraisemblance lié à l'intensité du processus, présenté par l'équation (2.7) et montrent des comportements divers et variés. Nous avons choisi de concentrer notre travail sur une sélection de quatre hashtags pour la suite de cette étude, chacun représentant une dynamique particulière liée à un sous-ensemble bien précis de nos données. Bien que nous ne présentions les résultats que pour ces quatre hashtags, l'ensemble du processus a été appliqué aux hashtags restants mais nous choisissons de ne présenter que les quatre hashtags sélectionnés afin d'analyser au mieux les résultats obtenus. Les hashtags considérés sont dès lors *#beach*, *#art*, *#cocktails* et *#Tomorrowland*. Ces quatre hashtags représentent respectivement les quatre catégories que nous retrouvions dans nos données au sein de la section 3.3: les hashtags dits "all time popular", les hashtags pour lesquels les données sont incomplètes, les hashtags possédant un comportement hebdomadaire et les hashtags à dynamique singulière, liés à un événement bien précis. La Figure 4.1 nous permet de voir les différents comportements présents. Un premier constat est que tous les comportements présentent un aspect oscillatoire, mais que celui-ci n'est pas d'amplitude constante ni de période fixe. L'absence des données concernant le hashtag *#art* se marque également dans l'estimation de l'infectiosité. Nous choisissons cependant de conserver ce hashtag pour la suite afin d'essayer malgré l'absence de données de proposer un modèle de prédiction viable pour celui-ci. Nous remarquons aussi qu'aucun de ces comportements n'est semblable à celui de son voisin. Certains présentent des dynamiques oscillatoires constantes, d'autres décroissantes, mais également des dynamiques quelconques.

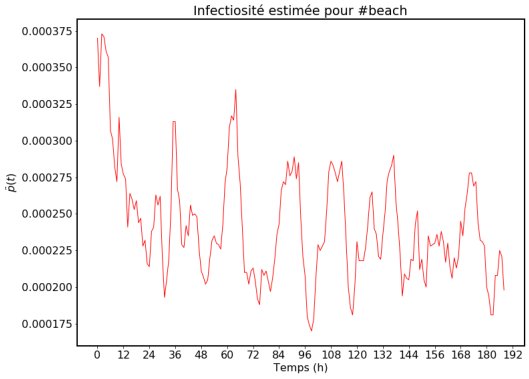
Notre première idée de modélisation était de proposer un modèle type à un nombre fini de paramètres et d'ajuster les paramètres de ceux-ci, comme ceci est présenté dans [14]. Cette idée imposait d'observer un comportement commun à toutes les infectiosités afin de proposer un modèle conçu comme un couplage de fonctions élémentaires (sinus, cosinus, exponentielles, logarithmes, polynômes, etc.). Malheureusement, comme les comportements présentent tous des caractéristiques différentes, cette idée est à abandonner et nous devons nous tourner vers une nouvelle méthodologie de modélisation de l'infectiosité. C'est cette méthodologie que nous allons maintenant présenter. Afin d'obtenir les résultats de cette section les codes présents en annexe G.2 ont été utilisés afin de calculer l'estimateur du maximum de vraisemblance et les codes de l'annexe D



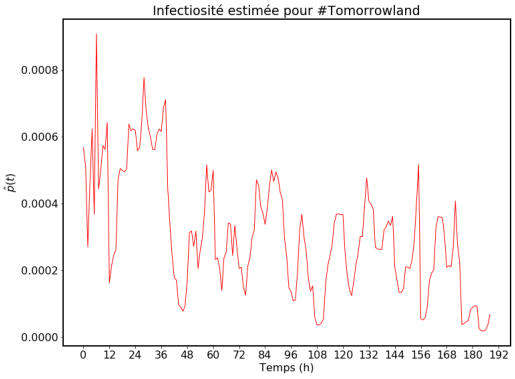
(a) *#cocktails*



(b) *#art*



(c) *#beach*



(d) *#Tomorrowland*

Figure 4.1: Estimateur du maximum de vraisemblance de l'infectiosité

ont permis l'affichage des données.

4.2 Modélisation par les séries de Fourier

Nous avons à plusieurs reprises rencontré les séries de Fourier lors de notre parcours universitaire en mathématiques. Nous allons nous baser sur la théorie des séries de Fourier afin de proposer une modélisation de l'infectiosité. Si nous nous référons à la théorie introduite par Fourier, nous savons que tout signal périodique peut, sous conditions de continuité et de dérivabilité, être réécrit sous la forme d'une suite de signaux sinusoïdaux. Plus précisément, si on considère un signal temporel $f(t)$ périodique de période T, on peut trouver un ensemble de coefficients a_n et b_n tels que

$$f(t) = a_0 + \sum_{i=1}^{+\infty} a_i \cos(\frac{2i\pi}{T}t) + b_i \sin(\frac{2i\pi}{T}t).$$

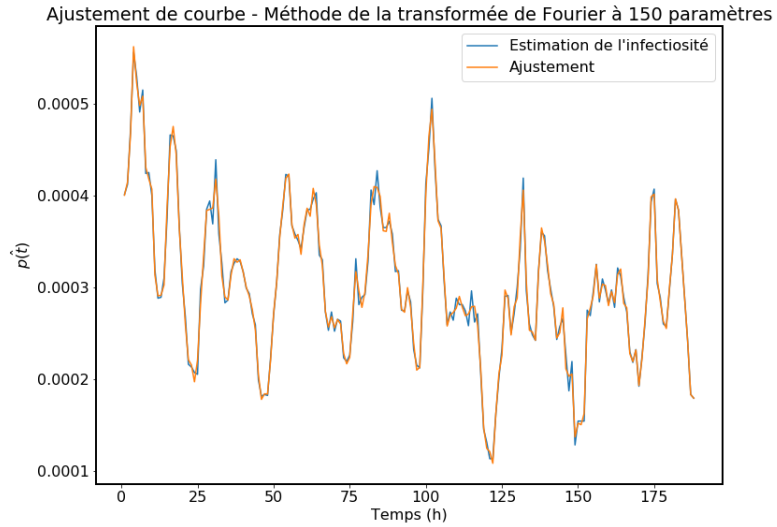
Nous pouvons donc, si nous choisissons de considérer l'infectiosité comme un signal périodique, tenter d'exprimer celle-ci comme une série de signaux sinusoïdaux. Plus précisément, et ce afin d'obtenir une expression finie, nous pouvons pour un entier k fixé considérer les k premiers termes de la décomposition de l'infectiosité en somme de signaux sinusoïdaux.

Nous pouvons dès lors proposer une modélisation de l'infectiosité comme étant une somme finie de cosinus. Notre motivation de ne considérer que les cosinus se base sur la volonté de créer un modèle qui reste le plus simple possible. L'infectiosité peut dès lors être déclinée comme

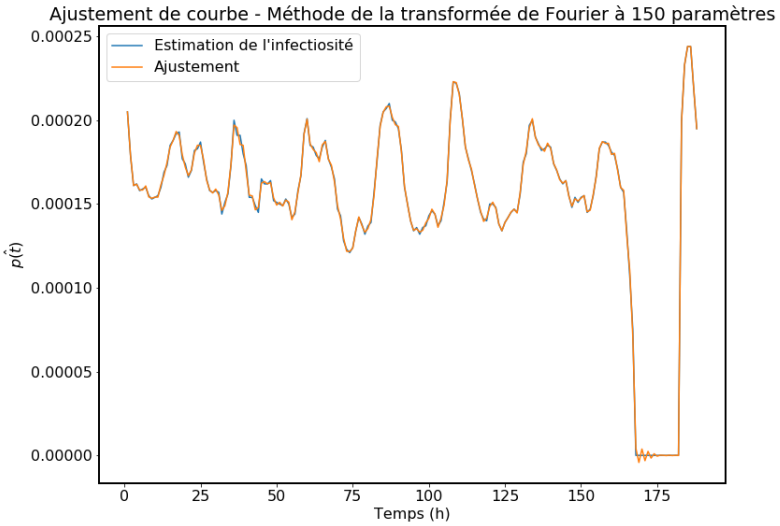
$$p(t) = a_0 + \sum_{i=1}^{k} a_i \cos(\frac{2i\pi t}{T}) \tag{4.1}$$

où T est la période, fixée dans notre cas à 168 heures (une semaine) et où k est l'ordre d'harmoniques considérées pour représenter le signal. Les coefficients a_i, $i = 0, ..., k$ sont déterminés en minimisant l'erreur des moindres carrés entre $p(t)$ et $\hat{p}(t)$ en utilisant la méthode de Levenberg-Marquardt telle que celle-ci est décrite dans la section 2.3.3.

La modélisation de $p(t)$ appliquée et optimisée pour un nombre d'harmoniques fixé à 150 nous donne les résultats présents aux Figures 4.2 et 4.3. Comme le présentent ces deux figures, l'écart entre les deux courbes est extrêmement faible pour les valeurs optimisées des paramètres. Ceci nous indique que notre choix de modèle s'avère être judicieux. Le

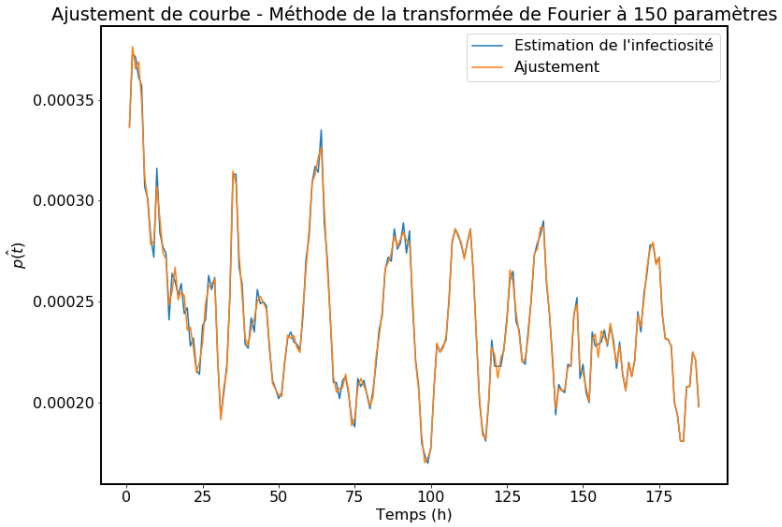


(a) *#cocktails*

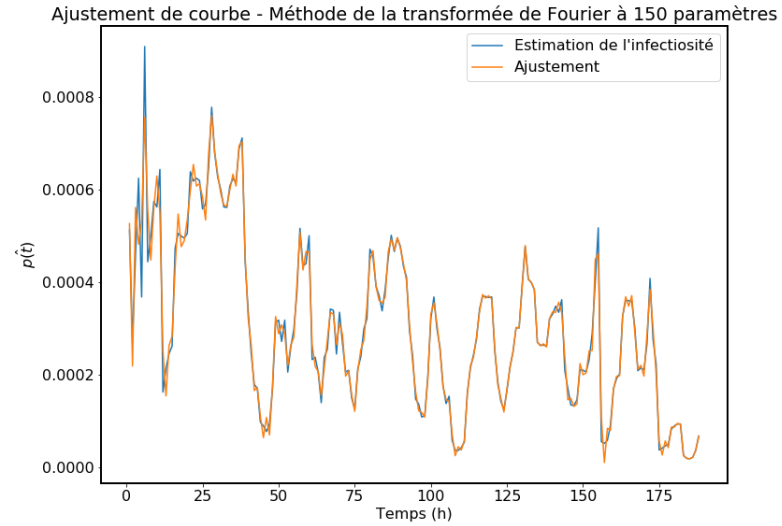


(b) *#art*

Figure 4.2: Ajustement de l'infectiosité, 150 termes considérés



(a) *#beach*



(b) *#Tomorrowland*

Figure 4.3: Ajustement de l'infectiosité, 150 termes considérés (2)

nombre de paramètres fixé à 150 permet à la courbe ajustée de coller assez fidèlement aux données empiriques.

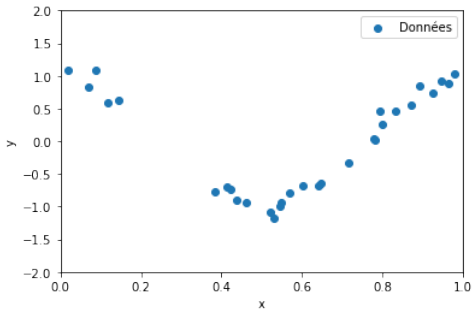
Nous n'allons cependant pas conserver cette modélisation, ou tout du moins, nous n'allons pas conserver ce nombre de paramètres pour la modélisation finale. Et pour cause: le modèle présenté colle tellement aux données qui est en réalité dans une situation connue dans le monde du machine learning comme de l'overfitting, ceci signifiant que le modèle est en quelque sorte trop habitué aux données et est donc incapable de faire une prédiction correcte. La section suivante va s'orienter vers un aspect machine learning de notre modélisation afin de proposer une méthode de construction considérant un nombre optimal de paramètres pour la série de Fourier.

4.3 Overfitting, underfitting et nombre idéal de paramètres

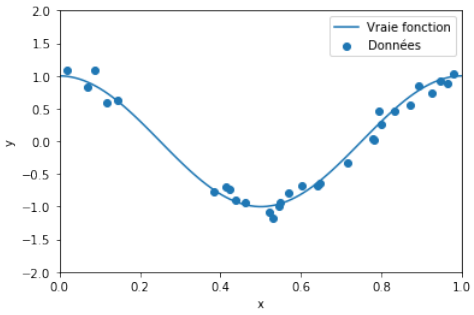
Nous allons maintenant brièvement présenter les concepts d'overfitting et d'underfitting et présenter en quoi ceux-ci ont affecté notre travail tout autant que la manière définitive que nous avons choisi pour modéliser l'infectiosité de notre processus. L'écriture de cet partie de notre travail se base sur la source [36] ainsi que sur les savoirs acquis lors du cours de Machine learning et Data Mining suivi à l'UNamur. Afin de présenter l'overfitting et l'underfitting, considérons l'ensemble de points (x_i, y_i) tels que présentés à la Figure 4.4a. Notre objectif est de construire un modèle qui soit une combinaison linéaire de monômes de degrés différents et qui tende à déceler le comportement réel induit par ces points. Les données sont issues de la fonction tracée à la Figure 4.4b mais du bruit a été ajouté à ces données.

Si nous tentons de modéliser ces données par un polynôme de degré 1, nous obtenons les résultats de la Figure 4.4c. Cette figure nous permet d'avoir une première approche de ce qu'est l'underfitting. Ce cas de figure présente un modèle qui n'a pas réussi à capturer le comportement réel sous-jacent des données. Le modèle est "trop simple" pour expliquer le comportement réel des données. Le modèle produit donc une situation d'underfitting i.e. il est trop général, trop peu complexe pour expliquer le comportement des données.

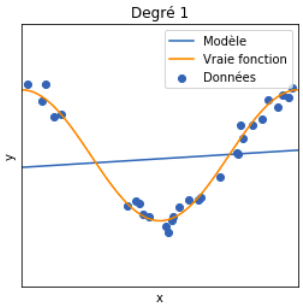
La Figure 4.4e présente quant à elle une modélisation de données via un polynôme de degré 15. Comme nous pouvons le remarquer dans cette représentation, l'écart entre ce modèle et l'ensemble de points est particulièrement faible. Le modèle colle donc extrêmement bien aux données. C'est précisément ce type de modèle que nous cherchons, le modèle qui produit le moins d'erreurs possible. Cependant, ce modèle n'est pas adapté à représenter le comportement réel sous-jacent dans nos données puisqu'il a capturé en



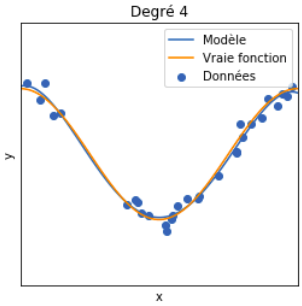
(a) Données



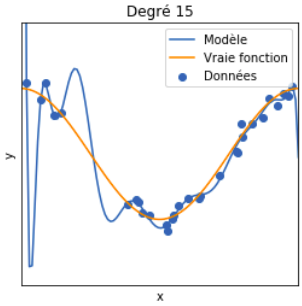
(b) Vraie fonction



(c) Approximation de degré 1



(d) Approximation de degré 4



(e) Approximation de degré 15

Figure 4.4: Exemple introductif à l'overfitting et l'underfitting. Source: [37]

s'entraînant sur les données le comportement induit par le bruit que nous avions ajouté. Nous nous trouvons ici dans une situation d'overfitting i.e. le modèle est trop complexe par rapport à la situation réelle et est trop habitué aux données. Si nous demandons à ce modèle d'estimer la valeur y d'un nouveau point x, il est fort probable que ce modèle produise un résultat éloigné de la valeur théoriquement vraie. Le modèle se généralise donc mal.

Quel est le modèle idéal ? Si les modèles trop peu complexes produisent des situations d'underfitting et le modèle trop complexes des situations d'overfitting, il semble évident que les modèles idéaux se trouvent entre ces deux cas. Afin de juger si la complexité du modèle est adaptée aux données, il convient de tester ce dernier. La procédure courante dans le machine learning est de scinder les données en deux parties distinctes appelées ensemble d'entraînement et ensemble de test. L'ensemble d'entraînement est, comme son nom l'indique, destiné à entraîner le modèle et à en ajuster les paramètres. L'ensemble de test sert quant à lui, une fois l'entraînement réalisé, à valider le modèle en évaluant l'erreur de prédiction produite par celui-ci. Le modèle idéal peut alors être choisi comme un modèle produisant des résultats de test et d'entraînement satisfaisants. Le choix d'un modèle ou d'un autre est donc affaire de compromis.

4.3.1 Utilisation des concepts machine learning

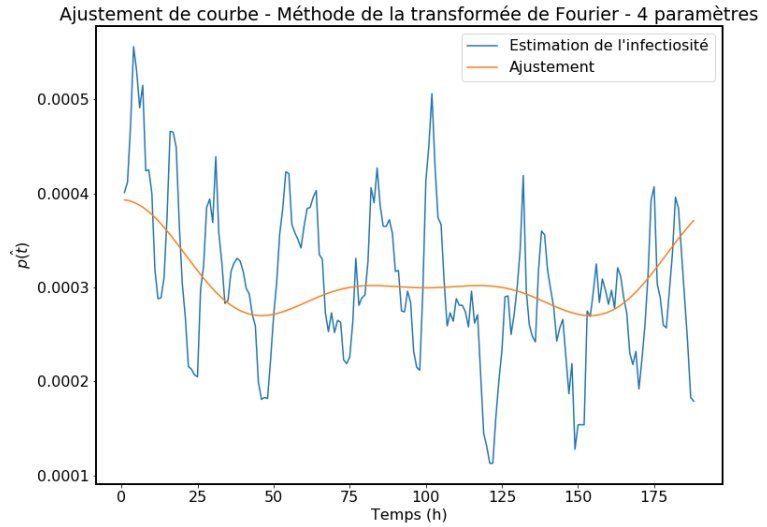
Nous allons maintenant détailler la méthodologie mise en place pour introduire ces concepts de machine learning dans notre parcours de modélisation de l'infectiosité. Dans un premier temps, comme présenté dans la section 4.2, nous remarquons qu'un modèle à 150 paramètres provoque une situation d'overfitting i.e. la série de Fourier à 150 termes cherchant à expliquer les données empiriques est trop habituée au jeu de données et n'arrive donc pas à capturer le réel comportement sous-jacent présent.

Notre première approche afin d'éviter l'overfitting du modèle a été de séparer les données en deux ensembles, un ensemble d'entraînement et un ensemble de test. Nous avons choisi de séparer les données en considérant un tiers des données pour le test et les deux tiers restants pour l'entraînement. Nous avons ensuite entraîné un ensemble de modèles dont le nombre de paramètres variait entre 2 et 124 paramètres sur le jeu de données d'entraînement. Suite à l'entraînement de ces données, nous avons testé les modèles sur l'ensemble de test pour évaluer l'erreur associée à chaque modèle. Notre première idée suite à l'entraînement et au test des modèles a été de considérer le modèle qui minimisait l'erreur de test sur le tiers de données liées au test.

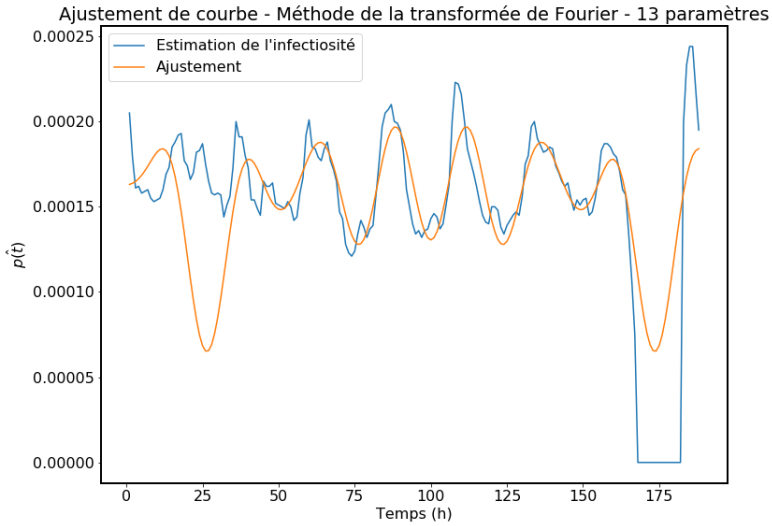
Suite à la mise en place de cette méthodologie, nous avons obtenu les fonctions d'infectiosité présentées aux Figures 4.5 et 4.6. Dans cette idée de minimisation de l'erreur d'entraînement, c'est le programme qui choisit lui même le nombre idéal de paramètres afin de minimiser l'erreur de test. Nous pouvons alors observer sur ces figures l'effet inverse de l'overfitting: l'underfitting. En effet, nous nous trouvons ici dans une situation où le modèle est devenu trop prudent, où il évite en quelque sorte de prendre des risques. Dans ce cas, le modèle est trop général pour représenter réellement les données, il n'a pas réussi à capturer le comportement sous-jacent des données. Notons cependant que la Figure 4.5b pour le hashtag $\#art$ semble malgré un ensemble de données manquantes relativement bien s'adapter aux données.

Nous nous trouvons donc entre deux situations. La première, où le choix du nombre de termes de la série de Fourier à considérer est laissé à la personne mettant en place la modélisation. Cette première option est à proscrire, et ce pour trois raisons principales. Les deux premières raisons sont l'overfitting et l'underfitting, qui peuvent survenir dans le cas où le choix de l'utilisateur n'est pas judicieux. La troisième raison, vue d'un point de vue un peu plus pratique, est l'absence d'automatisation de recherche du nombre de paramètres, ce qui est précisément le contraire de ce que nous cherchons à mettre en place. La seconde situation est de laisser le programme choisir de lui-même le nombre idéal de paramètres en fonction de l'erreur de test que le modèle produit. Ce choix n'est malheureusement, comme les montrent les Figures 4.5 et 4.6, pas judicieux. Il est dès lors légitime de se demander quelle méthode proposer pour que le modèle sélectionne automatiquement de lui-même un nombre adapté de termes afin de modéliser l'infectiosité.

La solution que nous proposons, et qui semble donner des résultats concluants pour l'ensemble des infectiosités liées à nos hashtags est de mélanger les deux options ci-dessus. L'idée qui sort de l'association des deux situations ci-dessus est la suivante: nous laissons le modèle décider du nombre idéal de paramètres en fonction de l'erreur de test que produisent ceux-ci, mais nous lui imposons cependant un nombre minimal de paramètres. Dans notre cas, imposer un nombre minimal de 15 paramètres (15 termes de la série de Fourier) offre des résultats intéressants et le code appliqué à l'ensemble des 91 infectiosités considérées permettait d'obtenir des résultats probants. La Figure 4.7 permet de visualiser les erreurs de test totales obtenues en fonction du nombre de paramètres considérés et présente clairement le nombre de paramètres réalisant l'erreur de test minimale. Comme le montrent les quatre sous-figures, ce nombre est variable d'un cas à l'autre et par conséquent une modélisation ne sera pas l'autre.

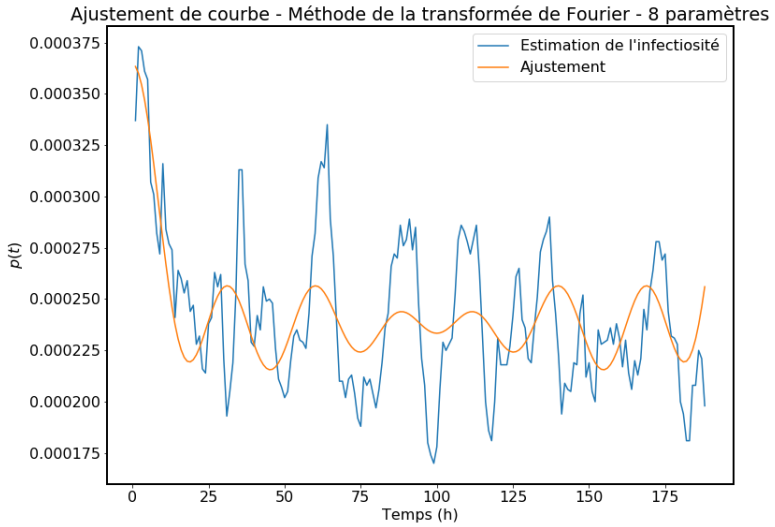


(a) *#cocktails*

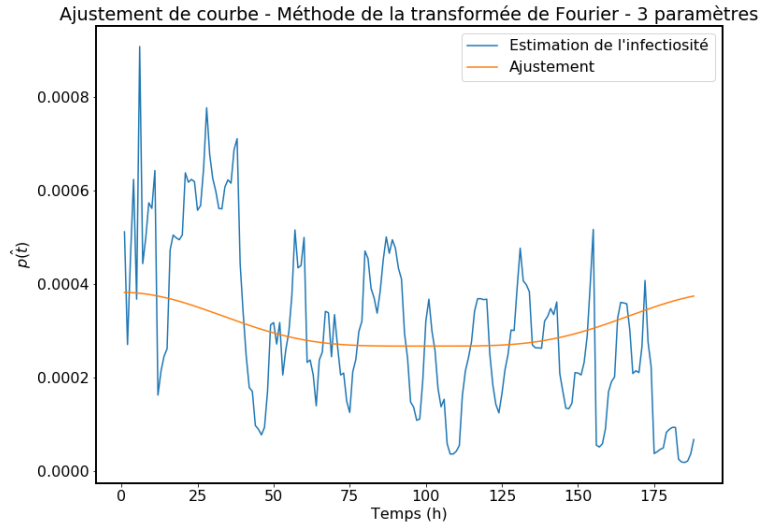


(b) *#art*

Figure 4.5: Ajustement de l'infectiosité, minimum sur l'ensemble de test

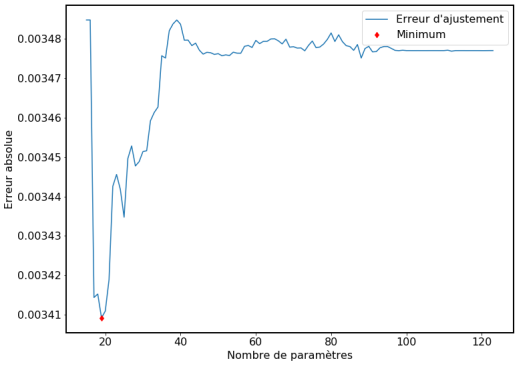


(a) *#beach*

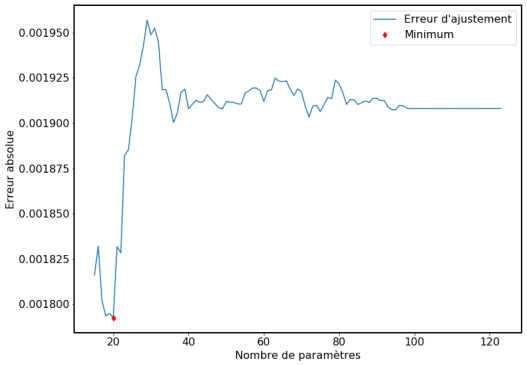


(b) *#Tomorrowland*

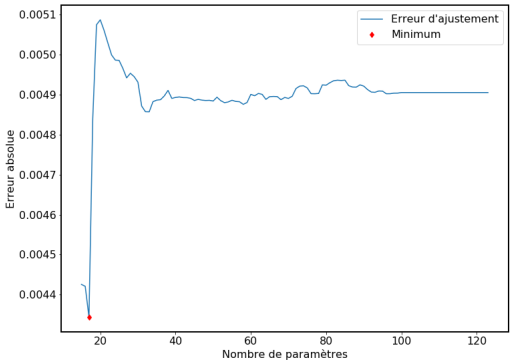
Figure 4.6: Ajustement de l'infectiosité, minimum sur l'ensemble de test (2)



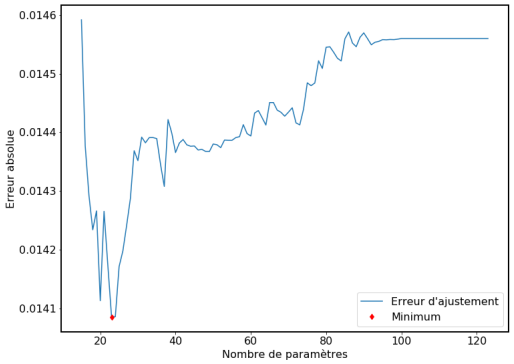
(a) Erreur de test pour *#art*



(b) Erreur de test pour *#beach*



(c) Erreur de test pour *#cocktails*

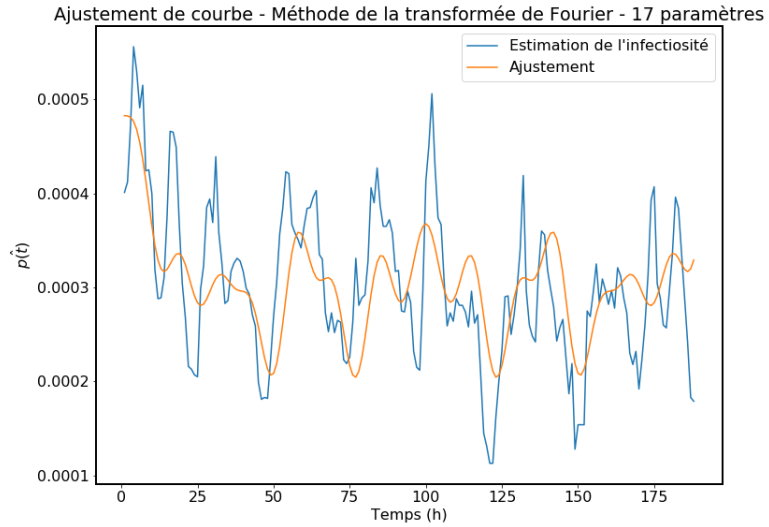


(d) Erreur de test pour *#Tomorrowland*

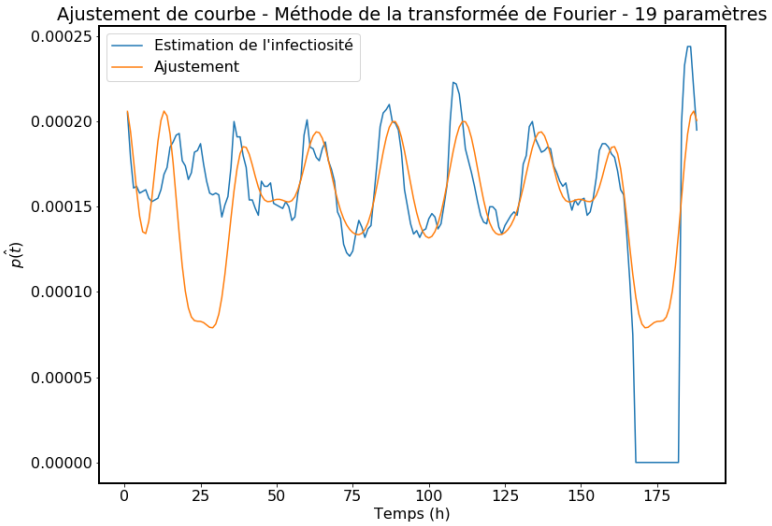
Figure 4.7: Erreurs de test en fonction du nombre de paramètres considérés

Les Figures 4.8 et 4.9 nous présentent les résultats en fonction du nombre optimal de paramètres. A l'exception de la Figure 4.8a pour laquelle l'ajustement ne coïncide que très peu avec l'infectiosité empirique, l'ensemble des courbes ajustées se rapproche assez fidèlement des courbes empiriques ce qui nous rassure sur notre choix d'imposer un nombre minimal de paramètres pour le modèle tout en sélectionnant le nombre minimisant l'erreur de test. Le résultat présenté pour *#art* nous conforte dans l'idée que ce modèle permet d'apporter de bons résultats même en présence d'un faible pourcentage de données manquantes.

Le résultat correspondant à *#cocktails* présent à la Figure 4.8a n'est que peu adapté à l'infectiosité empirique: déphasage, décalage d'amplitude, etc. Nous allons cependant conserver cette infectiosité afin d'observer les résultats que celle-ci peut produire dans le pattern de prédiction que nous présenterons dans le chapitre 5. Le code présenté dans l'annexe E représente la méthode finale d'ajustement utilisée.

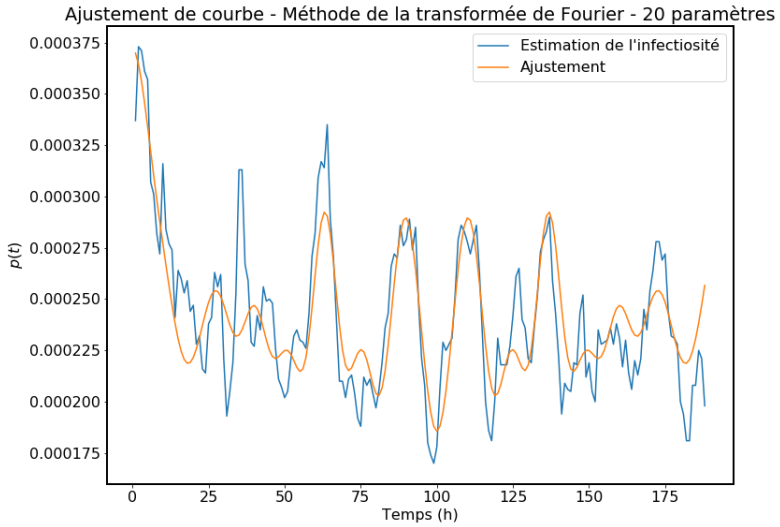


(a) Infectiosité ajustée pour *#cocktails*

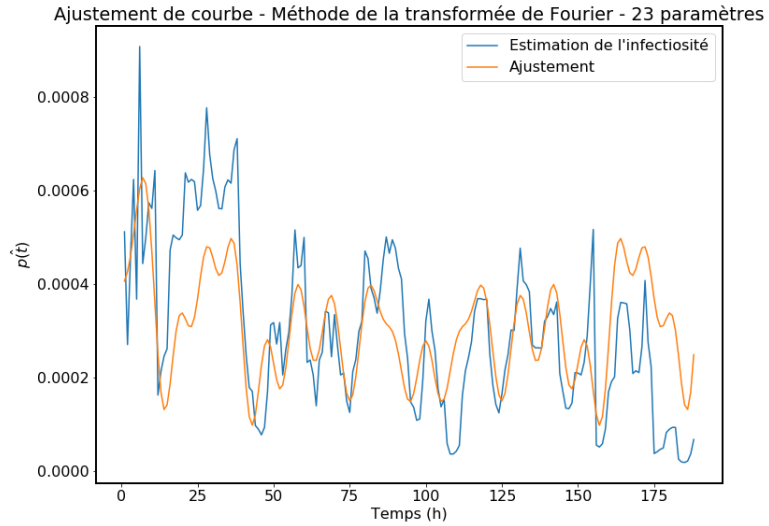


(b) Infectiosité ajustée pour *#art*

Figure 4.8: Ajustement de l'infectiosité, nombre idéal de paramètres



(a) Infectiosité ajustée pour *#beach*



(b) Infectiosité ajustée pour *#Tomorrowland*

Figure 4.9: Ajustement de l'infectiosité, nombre idéal de paramètres (2)

Chapitre 5

Prédiction de dynamique des hashtags

Dans ce dernier chapitre, le moment est venu de tester notre modèle afin d'évaluer si celui-ci produit des résultats de prédiction en adéquation avec les données observées. Nous pouvons avant de présenter les résultats obtenus rappeler la méthodologie mise en place afin d'évaluer l'activité future d'un hashtag. Dans un premier temps, et ce grâce aux données empiriques du hashtag, nous calculons l'estimateur du maximum de vraisemblance de l'infectiosité (section 2.2.2). Grâce à cette infectiosité empirique et à la théorie des séries de Fourier, nous proposons un modèle d'infectiosité sous forme de combinaison linéaire de signaux cosinusoïdaux et ajustons les coefficients de Fourier de ce modèle en minimisant l'erreur des moindres carrés entre le modèle et les données empiriques (sections 2.3.3 et 4.2). Une fois ce modèle calibré, nous calculons l'activité future du hashtag en calculant l'intensité du processus et en intégrant cette dernière (section 2.1). L'annexe G.3 présente le code utilisé afin de réaliser les prédictions d'activité et l'annexe F celui utilisé afin d'afficher la prédiction et les données empiriques ainsi que de calculer le score de la prédiction.

Afin d'évaluer la qualité de nos prédictions, nous nous baserons sur deux indicateurs. Le premier, intuitif, est la représentation graphique confrontant les données empiriques à l'activité future estimée par le modèle. Le second indicateur sera une métrique d'erreur notée ϵ, évaluée en calculant l'erreur absolue de la prédiction et en divisant cette dernière par le nombre total de réutilisations empiriques. Cette métrique sera donc positive et proche de 0 lorsque la prédiction est excellente. Nous appellerons également occasionnellement cette métrique score de prédiction ou encore plus simplement score. Nous avons choisi pour tester notre modèle de considérer 48 heures d'observation i.e. nous commençons à prédire l'activité future à partir de 48 heures et de nous arrêter après 168

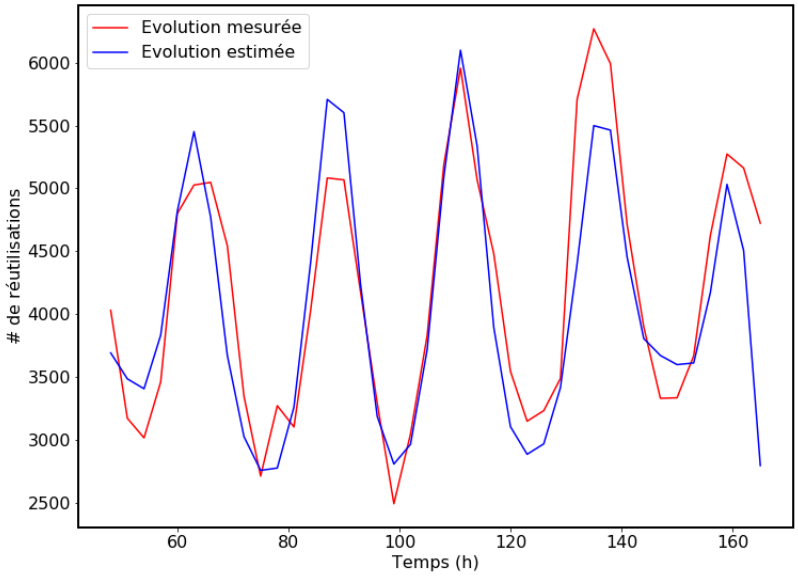


Figure 5.1: Prédiction d'activité future pour *#art*

heures. Nous proposons dès lors une prédiction sur 120 heures ce qui correspond à une période de 5 jours. La taille de fenêtre pour la prédiction a été fixée à 2 heures ce qui signifie que nous obtiendrons des résultats par intervalle de deux heures.

#art: Le hashtag *#art* est le premier auquel nous allons nous intéresser. La Figure 5.1 permet dans un premier temps d'observer la dynamique mesurée et la dynamique prédite par notre modèle. Nous pouvons immédiatement constater le bon fonctionnement du modèle sur ce premier cas. En effet, la dynamique oscillatoire du hashtag a été parfaitement capturée et les écarts entre la courbe de prédiction et la courbe empirique sont relativement petits. La mesure d'erreur de prédiction ϵ vaut pour ce hashtag $\epsilon = \frac{15288.224118}{167307} = 0.091378269397$. Cette valeur est assez faible et nous conforte dans l'idée que dans le cadre de ce hashtag, la modélisation s'avère adaptée.

Étant donné que nous avons conservé ce hashtag pour une raison bien précise, qui est

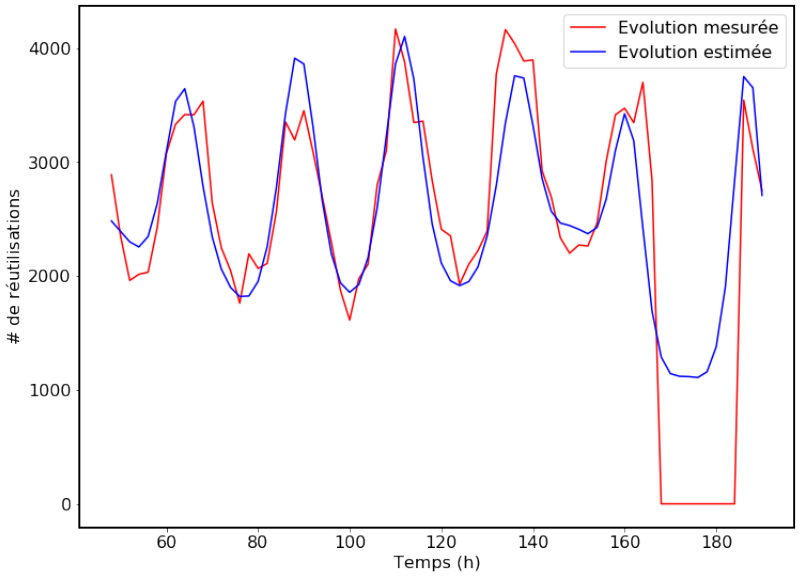


Figure 5.2: Prédiction d'activité future pour *#art* jusqu'à 192h

l'absence de récolte de données pendant un certain intervalle de temps, il semble judicieux d'essayer d'utiliser notre modèle sur un intervalle de temps plus grand, prenant en compte ce léger manque de données et d'observer l'erreur que produit ce modèle ainsi que la dynamique qu'il tente de prédire. La Figure 5.2 présente l'évolution réelle et prédite, qui semble au vu du reste de la dynamique être en adéquation avec les données observées (une dynamique oscillatoire, typique des hashtags dits "all time popular") bien que le creux présent à l'endroit du manque de données soit un peu plus prononcé que ceux du reste de la dynamique. La valeur de l'erreur de prédiction qui vaut $\epsilon = \frac{30109.479542}{176714} = 0.170385365857$ présage elle aussi que malgré l'absence de données, nous pouvons nous fier à cette prédiction et que le score de cette prédiction est admirable.

#beach: Le second hashtag auquel nous nous intéressons est *#beach*. Ce hashtag fait, tout comme le hashtag *#art*, partie des hashtags que nous avons qualifiés de "all time popular". La Figure 5.3 présente la dynamique d'évolution prédite comparée à la dynamique empirique. La dynamique prédite se rapproche globalement de la dynamique observée et ces deux-ci sont particulièrement proches sur les 72 premières heures considérées. Nous remarquons cependant un écart plus prononcé sur les heures suivantes mais cet écart n'évoque rien de dramatique. En ce qui concerne le score de la prédiction pour

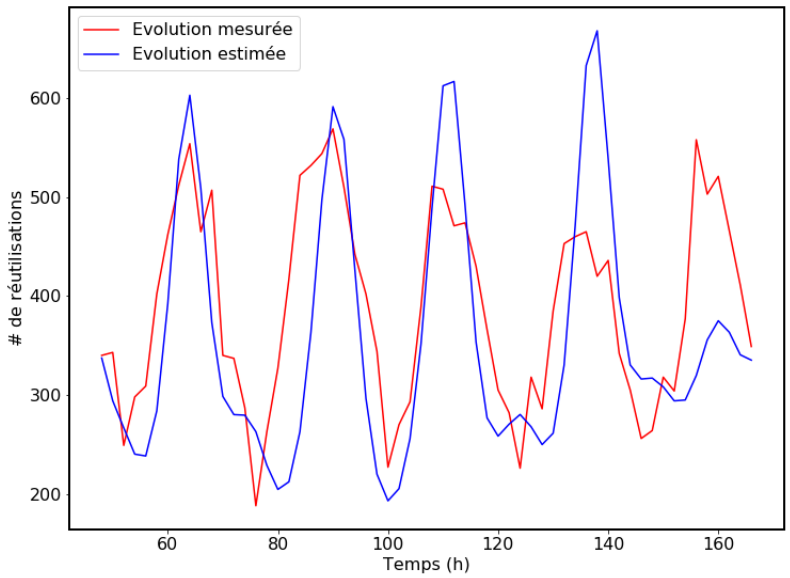


Figure 5.3: Prédiction d'activité future pour *#beach*

ce hashtag, il est de $\epsilon = \dfrac{4592.146128}{23414} = 0.196128219356$ qui est un score assez bon. Ces deux résultats combinés nous invitent à considérer que, dans le cadre de ce hashtag, la modélisation est adaptée.

#cocktails: Le troisième hashtag de notre sélection, pour lequel l'ajustement de l'infectiosité était moins précis que pour les trois autres considérés (Figure 4.8a), est le hashtag *#cocktails*. En effet, rappelons-nous que malgré le nombre optimal de paramètres sélectionnés par notre algorithme dans la série de Fourier, l'ajustement de l'infectiosité présentait des oscillations décalées par rapport aux oscillations observées et des amplitudes parfois inadaptées. Suite à l'exécution de l'algorithme de prédiction, nous obtenons les résultats présentés à la Figure 5.4 ainsi qu'un score de $\epsilon = \dfrac{1951.365409}{5102} = 0.382470679929$. Le score ainsi que la représentation des deux courbes montrent un résultat plus dégradé que ceux présentés pour les deux hashtags précédents. En effet, la figure présente dans un premier temps un bon départ de prédiction sur environ 48 heures qui se dégrade rapidement en produisant des déphasages légers et des amplitudes trop

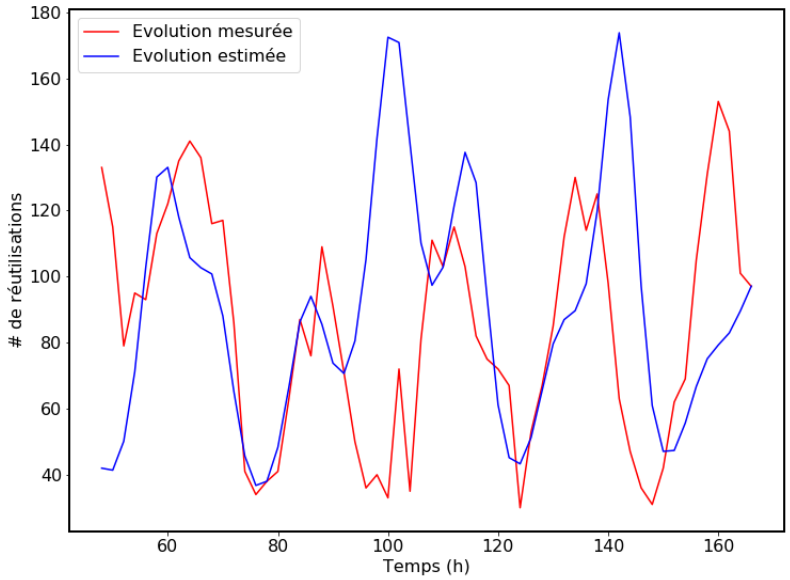


Figure 5.4: Prédiction d'activité future pour *#cocktails*

importantes en comparaison aux données observées. Le score de prédiction confirme ce déclin de précision, étant plus important que ceux des hashtags précédents.

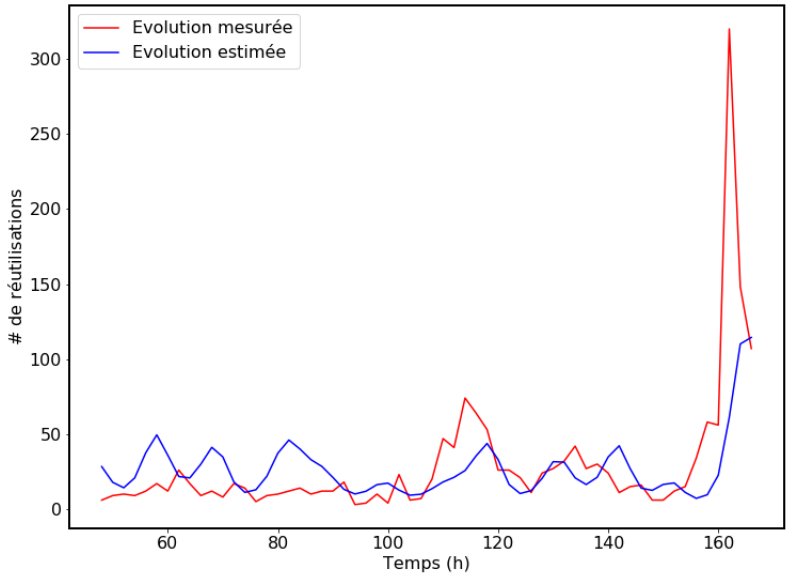
#Tomorrowland: Finalement, le dernier hashtag que nous avons analysé en profondeur est le hashtag *#Tomorrowland* qui est lié à un événement ponctuel. Malgré une infectiosité assez bien ajustée pour ce dernier, c'est malheureusement pour ce hashtag que nous obtenons les résultats les moins bons. En effet, sur la représentation graphique de la prédiction que nous pouvons retrouver à la Figure 5.5a montre un effet oscillatoire amplifié au début de la prédiction par rapport aux données ainsi qu'un décalage présent de manière constante entre la courbe de prédiction et la courbe de données empiriques. Le score associé à cette prédiction est de $\epsilon = \dfrac{1144.914429}{1730} = 0.661800247977$, ce qui est un résultat très pauvre et confirme notre intuition que dans le cadre de ce hashtag, le modèle n'était pas adapté à la prédiction de l'évolution de la dynamique. Nous nous sommes alors intéressés à observer cette évolution jusqu'à 192 heures ce qui nous a donné les résultats de la Figure 5.5b et un score de $\epsilon = 0.803571865414$. Nous remarquons, aussi bien en analysant le score que la représentation graphique confrontant la prédic-

tion et les données empiriques, que la situation se détériore si l'on tente de prédire sur un intervalle de temps. Les résultats sont similaires lorsque nous augmentons la durée d'observation (passage de 48 heures d'observation à 72 heures d'observation) et deviennent médiocres lorsque nous diminuons la durée d'observation sous 24 heures ($\epsilon > 1$). Comme nous allons le présenter ci-après, nous avons en réalité découvert que malgré une modélisation cohérente de l'infectiosité dans le cadre des hashtags liés à des événements ponctuels, notre modèle était globalement peu efficace pour évaluer l'activité future du hashtag sur le réseau.

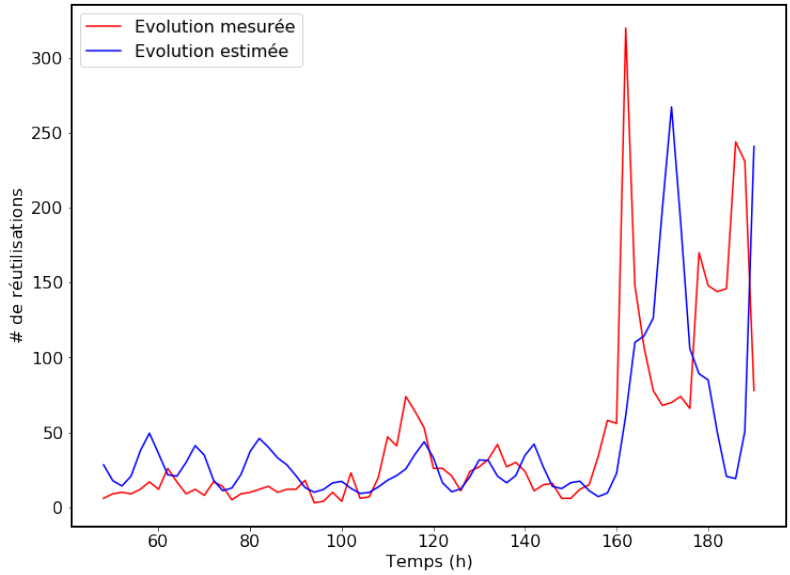
5.1 Résumé des résultats

Nous allons maintenant pour les 91 hashtags récoltés détailler à l'aide de la Table 5.1 les résultats obtenus. Pour ce faire, nous allons proposer d'associer un grade à chacune de nos prédictions en fonction de l'erreur de prédiction (score) que celles-ci ont obtenu. Nous proposons une séparation en quatre grades distincts, ceux-ci étant "bon", "moyen", "mauvais" et "médiocre" correspondant respectivement à des valeurs de ϵ contenues dans les intervalles $[0, 0.3]$, $]0.3, 0.5]$, $]0.5, 1[$ et $[1, +\infty[$. Nous présenterons également les résultats en marquant la distinction entre les différents types de dynamiques que nous avons décelées lors de l'analyse des données. Ces différentes dynamiques étaient pour rappel les hashtags "all time popular", les hashtags pour lesquels une partie de la récolte des données avait eu des ratés et les hashtags avec des dynamiques particulières au sein desquels nous avions décelé deux types de dynamiques distinctes: les dynamiques liées à des événements ponctuels et les dynamiques pouvant être expliquées par les différents jours de la semaine (voir section 3.3.1).

La Table 5.1 résume les résultats obtenus. Nous pouvons remarquer un pourcentage de "bonnes" prédictions total de 71,43 %, et si nous considérons également les hashtags ayant donné un score de prédiction moyen, ce pourcentage dépasse les 85%. Les deux hashtags pour lesquels une partie des données manquaient nous offrent tous deux de bonnes prédictions avec des scores inférieurs à 0.3. Nous déplorons cependant les résultats obtenus par le modèle appliqué aux hashtags à dynamique particulière, où celui-ci n'a malheureusement pas réussi à capturer la dynamique, ceci étant probablement dû à une modélisation erronée de l'infectiosité.



(a) Prédiction d'activité future pour *#Tomorrowland*, jusqu'à 168



(b) Prédiction d'activité future pour *#Tomorrowland*, jusqu'à 192

Figure 5.5: Prédiction d'activité future pour *#Tomorrowland*

	all time popular	évnt. ponctuel	données manq.	dyn. hebdo
bon	63 (69,23%)	0 (0%)	2 (2,2%)	0 (0%)
moyen	10 (10,98%)	2 (2,2%)	0 (0%)	2 (2,2%)
mauvais	2 (2,2%)	4 (4,4%)	0 (0%)	0 (0%)
médiocre	1 (1,09%)	5 (5,5%)	0 (0%)	0 (0%)

Table 5.1: Pourcentages de catégorie de prédiction par type de dynamique

Conclusion

En guise de conclusion, rappelons tout d'abord les étapes qui nous ont permis de mener notre travail à son terme. Dans un premier temps, nous nous sommes attelés à définir les concepts de base des réseaux sociaux, des médias sociaux ainsi que diverses interactions et contenus présents sur ces derniers afin que tout lecteur, du plus novice au plus aguerri, puisse poursuivre sans encombre la lecture de ce manuscrit. Nous avons également introduit les processus ponctuels afin de pouvoir définir les processus de Hawkes, sur lesquels l'intensité de notre processus de prédiction se basait.

La description de notre modèle a alors pris forme. Nous souhaitions construire un modèle simple permettant de modéliser la viralité et l'expansion d'un contenu sur les réseaux sociaux. Pour ce faire, en nous basant sur les processus de Hawkes, nous avons proposé une intensité de processus permettant de dégager, en un temps t donné, la probabilité d'obtenir une récidive d'activité en ce temps. Cette intensité, basée sur trois ingrédients (l'infectiosité, le nombre de followers et le noyau mémoriel), a été façonnée de manière à proposer un comportement de processus de Hawkes. Il nous a alors suffi d'utiliser cette modélisation de l'intensité et des outils adéquats afin de proposer une prédiction d'activité basée sur l'intégration de l'intensité. La forme du noyau mémoriel étant connue pour Twitter et le nombre de followers étant mesuré empiriquement, le paramètre le plus important de cette modélisation s'est avéré être le taux d'infection du processus. C'est pourquoi nous avons considéré la modélisation de celui-ci de manière indépendante et avons proposé l'association d'un panel de techniques pour le modéliser de la meilleure manière qui soit et que cette modélisation soit, qui plus est, automatisée.

Nous avons également présenté le processus suivi afin de capturer des données empiriques pour tester nos modèles et détaillé les différents comportements présents dans ces données. Tout ceci nous a permis de tester notre modèle, qui s'avère, si l'on considère les grades de scores proposés précédemment, être bon sur 71 % des cas que nous avons testés. De manière plus précise, en observant les résultats présentés au chapitre

5, nous pouvons remarquer que notre modèle est particulièrement adapté aux hashtags dits "all time popular" même si une faible quantité d'information est manquante. Nous déplorons cependant les résultats sur les hashtags à dynamique particulière. En effet, peu de prédictions offrent des résultats satisfaisants, et ce probablement dû à des taux d'infection plus complexes liés à ces dynamiques ou à l'absence de périodicité de ceux-ci.

Bien que nous offrions ici un détail complet sur la méthodologie de construction du modèle, nous sommes conscient que certains points pourraient jouir d'une amélioration, ce qui permettrait d'obtenir des résultats potentiellement plus précis. Notons par exemple une amélioration du noyau mémoriel. La forme de ce dernier a été développée via des données empiriques mais sans aucune connaissance de la structure du réseau. Nous pourrions envisager de renforcer la modélisation de ce dernier en obtenant plus d'informations sur le réseau ou éventuellement en modélisant le réseau via un graphe de configuration. Un autre aspect de notre travail qui aurait pu apporter plus de précision à la sélection de paramètres est d'utiliser une validation croisée sur notre modèle afin d'obtenir plus d'informations sur la fiabilité de celui-ci. Nous avons également choisi de ne conserver que les termes en cosinus dans la modélisation de l'infectiosité, ce qui force celle-ci à avoir un comportement pair. C'est une hypothèse très lourde mais qui nous permettait d'obtenir des résultats satisfaisants sans considérer de trop grands nombres de termes dans la série de Fourier. Un autre aspect qui pourrait être examiné serait de considérer une série de Fourier prenant en compte également les harmoniques de sinus afin de voir si les résultats ne se trouvent pas améliorés par cette modélisation.

Nous sommes parvenus au terme de ce travail à proposer une méthodologie de prédiction de popularité des contenus sur les réseaux sociaux - tout du moins pour les hashtags dans le cas de Twitter - ce qui était notre objectif. Bien que certaines améliorations soient encore possibles pour ce travail, un grand pas a été réalisé, proposant une méthode automatisée de modélisation d'infectiosité quel que soit le cas analysé et permettant ainsi d'accroître le nombre de cas à traiter.

Références

[1] Le Blog du modérateur, *Les 50 chiffres à connaître sur les médias sociaux en 2018*, https://www.blogdumoderateur.com/50-chiffres-medias-sociaux-2018/, consulté le 10/08/2018

[2] Mercator Publicitor, *Définition de "Réseaux Sociaux"*, http://www.mercator-publicitor.fr/lexique-marketing-definition-reseaux-sociaux, consulté le 28/07/2017

[3] L'internaute, *Définition de "Réseau Social"*, http://www.linternaute.com/dictionnaire/fr/definition/reseau-social, consulté le 28/07/2017

[4] Futura-Sciences, *Définition de "Réseau Social"*, http://www.futura-sciences.com/tech/definitions/informatique-reseau-social-10255/, consulté le 28/07/2017

[5] Université de Namur, WebCampus, http://webcampus.unamur.be, consulté le 28/07/2017

[6] Edunow, *World's 10 largest databases in the world*, http://www.edunow.ga/2018/01/top-10-largest-databases-in-world-2018.html, Janvier 2018, consulté le 25/07/2018

[7] Justin Cheng et al., *Can cascades be predicted?*, 23rd International World Wide Web Conference, 2014

[8] Tsur O., Rappoport A., *What's in a Hashtag? Content based prediction of the spread of ideas in microblogging communities*, Web Search and Data Mining Conference, 2012

[9] Jaulin F., *Processus ponctuels Markoviens*https://www.lpsm.paris/pageperso/mazliak/Jaulin.pdf, 2008, consulté le 11/08/2018

[10] Laub, P. J. and Taimre, T. and Pollett, P. K., *Hawkes Processes*, ArXiv e-prints, 2015, `https://arxiv.org/abs/1507.02822`, consulté le 24 jullet 2018

[11] Schoenberg F., *Introduction to point processes*, Wiley Encyclopedia of Operations Research and Management Science, 2010

[12] Brillinger, D. R., Guttorp, P. M., and Schoenberg, F. P. *Point Processes, Temporal.* Encyclopedia of Environments 3, 2002, p. 1577-1581

[13] Qingyuan Zhao et al., *SEISMIC: A Self-Exciting Point Process Model for Predicting Tweet Popularity*, International Conference on Knowledge Discovery and Data Mining (KDD), 2015

[14] Ryota Kobayashi, Renaud Lambiotte, *TiDeH: Time-Dependent Hawkes Process for Predicting Retweet Dynamics*, 10th International AAAI Conference on Web and Social Media, April 2016

[15] Kenney J. F., Keeping E. S. *Moving Averages*, Mathematics of Statistics, Pt. 1, 3rd ed. Princeton, 1962, p. 221-223.

[16] Whittaker, E. T., Robinson, G. *Graduation, or the Smoothing of Data* The Calculus of Observations: A Treatise on Numerical Mathematics, 4th ed. New York: Dover, pp. 285-316, 1967

[17] R. Crane, D. Sornette, *Robust dynamic classes revealed by measuring the response function of a social system*, Proceedings of the National Academy of Sciences Oct 2008

[18] A.-L. Barabasi, *The origin of bursts and heavy tails in human dynamics*, Nature, 435:207, 2005

[19] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical recipes in C, Vol. 2*, Cambridge university press Cambridge, 1996, p.130-136, 788-796

[20] Méthode des trapèzes, `https://blogdemaths.wordpress.com/2015/07/26/ce-chercheur-qui-reinventa-la-roue-sans-le-savoir/#jp-carousel-3423`, consulté le 23 juillet 2018

[21] Marquardt, D., *An Algorithm for Least-Squares Estimation of Nonlinear Parameters*, SIAM Journal on Applied Mathematics 11, 1963, p. 431–441.

[22] Gavin H., *The Levenberg-Marquardt method for nonlinear least squares curve-fitting problems*, Department of Civil and Environmental Engineering, Duke University, 2017, p. 1–6

[23] Madsen K., Bruun Nielsen H., Tingleff O., *Methods for Non-Linear Least Squares Problems (2nd ed.)*, Informatics and Mathematical Modelling, Technical University of Denmark, DTU, 2004, p. 5-28

[24] Zinn-Bjorkman L., *Numerical Optimization using the Levenberg-Marquardt Algorithm*, `http://mads.lanl.gov/presentations/Leif_LM_presentation_m.pdf`, consulté le 23/07/2018

[25] Jensen T. C., *Les API pour les nuls, Edition limitée IBM*, John Wiley & Sons, Inc., 2015, p. 5-8

[26] Cooksey B., *An introduction to APIs*, Zapier, Inc., 2014, p. 5-8

[27] 3scale, *What is an API ?*, `https://www.3scale.net/wp-content/uploads/2012/06/What-is-an-API-1.0.pdf`, consulté le 26 juillet 2018

[28] Twitter Search API, `https://developer.twitter.com/en/docs/tweets/search/api-reference/get-search-tweets.html`, consulté le 26 juillet 2018

[29] Littman J., *Where to get Twitter data for academic research*, `https://gwu-libraries.github.io/sfm-ui/posts/2017-09-14-twitter-data`, consulté le 26 juillet 2018

[30] Twitter API Reference, `https://developer.twitter.com/en/docs/tweets/search/api-reference/premium-search#CountsEndpoint`, consulté le 25/07/2018

[31] Statistica, *Leading countries based on number of Twitter users as of April 2018*, `https://www.statista.com/statistics/242606/number-of-active-twitter-users-in-selected-countries/`, consulté le 26 juillet 2018

[32] Shortstack, *158 most popular hashtags for Instagram marketing and more.*, `https://www.shortstack.com/blog/158-most-popular-hashtags-for-instagram-marketing-and-more-2017/`, consulté le 26/07/2018

[33] Top Hashtags' Twitter account, https://twitter.com/tophashtags, consulté le 17 juillet 2018

[34] Junkee, *What is #MyHealthRecord And Why Is Everyone Talking About It?*, http://junkee.com/junk-explained-myhealthrecord/168004, consulté le 26 juillet 2018

[35] Australian Government, *My Health Record*, https://www.myhealthrecord.gov.au/news/media-release-my-health-record-opt-out-date-announced, consulté le 26 juillet 2018

[36] Shalev-Shwartz S., Ben-David S., *Understanding Machine Learning: From Theory to Algorithms*, Cambridge University Press, 2014, p.93-96

[37] Scikit Learn, *Underfitting vs. Overfitting*, http://scikit-learn.org/0.15/auto_examples/plot_underfitting_overfitting.html, consulté le 10 août 2018

Annexes

A Procédure de récupération de données sur Twitter

```
import tweepy
import csv
import time
import datetime
import numpy as np
import simplejson

# Collect tweet containing a given hashtag
def hashtag_collect(hashtag_name):
    start_time = time.time()

    consumer_key=''
    consumer_secret=''
    access_token=''
    access_token_secret=''

    auth = tweepy.OAuthHandler(consumer_key, consumer_secret)
    auth.set_access_token(access_token, access_token_secret)
    api = tweepy.API(auth,wait_on_rate_limit=True,
        ↪ wait_on_rate_limit_notify=True)

    # Open/Create a file to append data
    csvFile = open('data_collect.csv', 'a')
    time_followers = open(('collect_17-07/time_and_followers_%s.csv' %
        ↪ hashtag_name), 'a')
    #Use csv Writer
    csvWriter = csv.writer(csvFile)
    csvWriter_data= csv.writer(time_followers)
    csvWriter_data.writerow(['timestamps','followers'])

    print(hashtag_name)
    dates=['2018-07-30','2018-07-31','2018-08-01','2018-08-02','
        ↪ 2018-08-03','2018-08-04','2018-08-05','2018-08-06','2018-08-07
        ↪ ']
    for i in range(0,len(dates)-1,1):
```

```
        print(i,dates[i])
        for tweet in tweepy.Cursor(api.search,q=hashtag_name,count=200,
            ↪ since=dates[i],until=dates[i+1], lang='en').items():
                csvWriter.writerow([tweet.created_at, tweet.text.encode('
                    ↪ utf-8'),tweet.user.followers_count, time.mktime(
                    ↪ time.strptime(str(tweet.created_at),"%Y-%m-%d␣%H:%M
                    ↪ :%S"))])
                csvWriter_data.writerow([time.mktime(time.strptime(str(
                    ↪ tweet.created_at),"%Y-%m-%d␣%H:%M:%S")),tweet.user.
                    ↪ followers_count])
```

B Procédures de prétraitement et d'affichage des données (Python)

```
import csv
import pandas as pd
import time
import datetime
import numpy as np
import matplotlib.pyplot as plt

fig_size = plt.rcParams["figure.figsize"]
fig_size[0] = 12
fig_size[1] = 9
plt.rcParams["figure.figsize"] = fig_size
plt.rcParams['axes.linewidth'] = 2
plt.rcParams.update({'font.size': 16})

def timestamp_to_decimal_hour(t):
    t_hms=datetime.datetime.fromtimestamp(int(t)).strftime('%H:%M:%S')
    (h, m, s) = t_hms.split(':')
    return (int(h) * 3600 + int(m) * 60 + int(s))/float(3600)

def roundint(value, base):
    return int(value) - int(value) % int(base)
```

```
def raw_to_tideh(hashtag_name):
    dataframe=pd.read_csv(('Raw_data/time_and_followers_%s.csv'
                           %hashtag_name), index_col=None)
    sorted_df=dataframe.sort_values(by=['timestamps'], ascending=True)
    sorted_df=sorted_df.reset_index(drop=True)
    init_time=sorted_df.at[0,'timestamps']
    total_rows = sorted_df.shape[0]
    sorted_df['timestamps']= round((sorted_df['timestamps']-init_time)
                                              /float(3600),6)
    sorted_df.loc[-1] = [int(total_rows),
              float(timestamp_to_decimal_hour(init_time))]
    sorted_df.index = sorted_df.index + 1
    sorted_df.sort_index(inplace=True)
    sorted_df.loc[:, 'followers'] = sorted_df['followers'].apply(int)
    sorted_df.to_csv(('Tideh_format/tideh_format_%s.csv' % hashtag_name)
       ↪ ,header=False, index=False, sep="␣")


def histo_12to1(hashtag_name):
    dataframe=pd.read_csv(('Tideh_format/tideh_format_%s.csv'
    % hashtag_name),sep='␣',skiprows=1,header=None)
    df=dataframe.as_matrix()
    temps=df[:,0]
    followers=df[:,1]

    bins=range(0,roundint(np.max(temps)+24,12),12)
    fig=plt.figure()
    plt.hist(temps, bins=bins,rwidth=0.97, label='Average:␣%d'
                                  % np.average(followers))
    plt.title("Distribution␣des␣hashtags␣pour␣%s"%hashtag_name)
    plt.xlabel("Temps␣(h)")
    plt.ylabel("Frequence")
    plt.xticks(np.arange(int(np.min(temps)), int(np.max(temps))+12, 12))
    plt.savefig("Histogrames/histo12_%s.png"%hashtag_name, format='png')
    plt.close()

    bins=range(0,roundint(np.max(temps)+24,12),1)
```

```
    plt.figure()
    plt.hist(temps, bins=bins,rwidth=1.)
    plt.title("Distribution␣des␣hashtags␣pour␣%s"%hashtag_name)
    plt.xlabel("Temps␣(h)")
    plt.ylabel("Frequence")
    plt.xticks(np.arange(int(np.min(temps)), int(np.max(temps))+12, 12))
    plt.savefig("Histogrames/histo1_%s.png"%hashtag_name, format='png')
    plt.close()
```

C Procédure d'affichage des moyennes glissantes (Python)

```
import csv
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
fig_size = plt.rcParams["figure.figsize"]
fig_size[0] = 12
fig_size[1] = 9
plt.rcParams["figure.figsize"] = fig_size
plt.rcParams['axes.linewidth'] = 2
plt.rcParams.update({'font.size': 16})

def running_mean(x, N):
    cumsum = np.cumsum(np.insert(x, 0, 0))
    return (cumsum[N:] - cumsum[:-N]) / float(N)

df=pd.read_csv('moving_averages_exemple.csv', index_col=None,
              names=['temps','followers'], header=1)

temps=df.as_matrix(columns=['temps'])
followers=df.as_matrix(columns=['followers'])
followers=np.asarray(followers)

S_4=running_mean(followers, 4)
S_10=running_mean(followers, 10)
S_30=running_mean(followers, 30)
```

```
S_100=running_mean(followers, 100)

plt.plot (followers,'g-',linewidth=1, label='Donnees brutes')
plt.plot(S_4, 'r-',linewidth=1,label='Moyenne glissante, k=4')
plt.plot(S_10, 'b-',linewidth=1,label='Moyenne glissante, k=10')
plt.plot(S_30, 'm-',linewidth=1,label='Moyenne glissante, k=30')
plt.legend(loc='best')
plt.show()

plt.semilogy (followers,'g-',linewidth=1,label='Donnees brutes')
plt.semilogy(S_4, 'r-',linewidth=1, label='Moyenne glissante, k=4')
plt.semilogy(S_10, 'b-',linewidth=1,label='Moyenne glissante, k=10')
plt.semilogy(S_30, 'm-',linewidth=1,label='Moyenne glissante, k=30')
plt.legend(loc='best')
plt.show()
```

D Procédure d'affichage de l'infectiosité (Python)

```
import csv
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

fig_size = plt.rcParams["figure.figsize"]
fig_size[0] = 12
fig_size[1] = 9
plt.rcParams["figure.figsize"] = fig_size
plt.rcParams['axes.linewidth'] = 2
plt.rcParams.update({'font.size': 16})

def plot_infect(hashtag_name):
    df=pd.read_csv('Infectiosite/f_est_mv_%s.txt'%hashtag_name,
                   index_col=None, names=['temps','pt'])
    df.temps = df.temps.astype(int)

    temps=df.as_matrix(columns=['temps'])
```

```
    taux_infec=df.as_matrix(columns=['pt'])

    plt.figure()
    plt.plot(temps, taux_infec, 'r-',linewidth=1, label='test')
    plt.title('Infectiosite␣estimee␣pour␣%s'%hashtag_name)
    plt.xlabel("Temps␣(h)")
    plt.ylabel("$\hat{p}(t)$")
    plt.xticks(np.arange(np.min(temps), np.max(temps)+8, 12))
    plt.savefig("Plot_infectiosite/infectiosite_estimee_%s.png"
                %hashtag_name, format='png')
    plt.close()
```

E Procédure d'ajustement de l'infectiosité (Python)

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.optimize import curve_fit
import os
fig_size = plt.rcParams["figure.figsize"]

fig_size[0] = 12
fig_size[1] = 9
plt.rcParams["figure.figsize"] = fig_size
plt.rcParams['axes.linewidth'] = 2
plt.rcParams.update({'font.size': 16})

coefs_min=15


#split array in 2 inequal parts, 2/3 (for train) and 1/3 (for test)
def split_array(array):
    length = len(array)
    split_len=length/3.
    part1_len=2*split_len
    part1_len=int(round(part1_len))
    splitted_part1,splitted_part2=np.split(array, [part1_len])
```

```
    return splitted_part1, splitted_part2

#function to fit
def fourier(x, *a):
    ret = a[0]+(a[1] * np.cos((2*np.pi / tau) * x))
    for deg in range(2, len(a)):
        ret += (a[deg] * np.cos(((deg) * 2*np.pi / tau) * x))
    return ret

def fit_train(t,infectiosite,tau,k):

    coefs, coefs_cov = curve_fit(fourier, t, infectiosite, [1.0]*k,
       ↪ method='lm')

    error=0
    error_vect=list()
    for i in range(0,len(t)):
        error+=np.abs(infectiosite[i]-fourier(t[i],*coefs))
        error_vect.append(np.abs(infectiosite[i]-fourier(t[i],*coefs)))


    return coefs, error_vect, error


def fit_test(temps, infectiosite, *coefs):

    error=0
    error_test=list()
    for i in range(0,len(temps)-1):
        error+=np.abs(infectiosite[i]-fourier(temps[i],*coefs))
        error_test.append(np.abs(infectiosite[i]-fourier(temps[i],*coefs)
           ↪ ))

    return error

def coefs_selection(hashtag_name):
    filename = ('Infectiosite/f_est_mv_%s.txt' %hashtag_name)
```

```
t, infectiosite = np.loadtxt(filename,delimiter=',', unpack=True,
    ↪ skiprows=1)
tau=168

t_train,t_test=split_array(t)
infectio_train, infectio_test=split_array(infectiosite)
error_vector_test=list()

for i in range(coefs_min,len(t_train)-1):
    fitted_coefs,error_vector_train, error_train=fit_train(t_train,
        ↪ infectio_train, tau, i)
    test_error=fit_test(t_test, infectio_test,*fitted_coefs)
    error_vector_test.append(test_error)



if(np.argmin(error_vector_test)==0 or np.argmin(error_vector_test)
    ↪ ==1):
    error_vector_test[0]=np.max(error_vector_test)
    error_vector_test[1]=np.max(error_vector_test)

fig=plt.figure()
plt.plot(range(coefs_min,len(t_train)-1),error_vector_test, label='
    ↪ Erreur␣d\'ajustement')
plt.plot(np.argmin(error_vector_test)+coefs_min, error_vector_test[
    ↪ np.argmin(error_vector_test)], 'dr',label='Minimum')
plt.xlabel("Nombre␣de␣parametres")
plt.ylabel("Erreur␣absolue")
plt.legend(loc='best')
plt.savefig("erreur_fit/erreurfit_%s.png"%hashtag_name, format='png'
    ↪ )
plt.show()
plt.close(fig)


return error_vector_test, np.argmin(error_vector_test)
```

```
def write_coefs(hashtag_name):
    filename = ('Infectiosite/f_est_mv_%s.txt' %hashtag_name)
    t, infectiosite = np.loadtxt(filename,delimiter=',', unpack=True,
        ↪ skiprows=1)
    tau=168

    temp,argmin=coefs_selection(hashtag_name)
    coefs,a,b=fit_train(t,infectiosite,tau,argmin+coefs_min)
    np.savetxt('Fitted_params/parametres_%s.txt'%hashtag_name, coefs,
        ↪ delimiter=' ', fmt='%5.16f')

    nb_coefs = len(coefs)
    with open('Fitted_params/parametres_%s.txt'%hashtag_name, 'r+') as f
        ↪ :
        file_data = f.read()
        f.seek(0, 0)
        f.write(str(nb_coefs) + '\n' + file_data)

    fig = plt.figure()
    p1, = plt.plot(t,infectiosite, label='Estimation de l\'infectiosite'
        ↪ )
    p2, = plt.plot(t, fourier(t, *coefs), label='Ajustement')
    plt.title("Ajustement de courbe - Methode de la transformee de
        ↪ Fourier - %i parametres" %(len(coefs)))
    plt.xlabel("Temps (h)")
    plt.ylabel("$\hat{p(t)}$")
    plt.legend(loc='best')
    plt.savefig("fit_best/fit_best_fourier_%s.png"%hashtag_name, format=
        ↪ 'png')
    plt.show()
    plt.close(fig)

    return coefs
```

F Procédure d'affichage de la prédiction (Python)

```
import tweepy
import csv
import pandas as pd
import time
import datetime
import numpy as np
import matplotlib.pyplot as plt
import simplejson

fig_size = plt.rcParams["figure.figsize"]
fig_size[0] = 12
fig_size[1] = 9
plt.rcParams["figure.figsize"] = fig_size
plt.rcParams['axes.linewidth'] = 2
plt.rcParams.update({'font.size': 16})

def plot_activity(hashtag):
    dataframe=pd.read_csv(('/home/pierre/Bureau/Lien␣vers␣Memoire/Code␣
        ↪ 2-07␣fonctionnel/Run/prediction_%s.txt'%hashtag), index_col=
        ↪ None, header=None, sep='␣')
    df=dataframe.as_matrix()
    temps=df[:,0]
    real=df[:,1]
    predict=df[:,2]

    fig=plt.figure()
    plt.plot(temps, real, 'r-', label='Evolution␣mesuree')
    plt.plot(temps,predict, '-b', label='Evolution␣estimee')
    plt.legend(loc='best')
    plt.xlabel("Temps␣(h)")
    plt.ylabel("$\#$␣de␣reutilisations")
    plt.savefig("Prediction/predict_%s.png"%hashtag, format='png')
    plt.show()
    plt.close(fig)
```

```
    sum(abs(real-predict))
    sum(real)
    error=sum(abs(real-predict))/sum(real)
    print("Score␣=",error)
```

G Codes C

Les codes présentés dans cette annexe sont des adaptations du code fourni en rapport avec [14] afin d'être utilisés dans le cadre de notre étude. Ils se basent très fortement sur la version originale de ceux-ci.

G.1 Header et sous-routines

Header

```
#include <stdio.h>
#include <math.h>
#include <stdlib.h>
#include <string.h>

#define DT 0.1 // Time Step for solving integral eq. (hour)
#define Th -0.242 // Theta: Kernel parameter
#define WIN 4 // Window for estimating p_T
#define DIM 4 // # parameters: p(t)

extern double nb_params;


struct data_rt{ // Struct: Retweet Data
  int num; // #Events
  int *n_i; // #Followers of each retweetor
  double *t_sp; // Time of each retweet
};

// Memory kernel: phi_t and its integral: h_t.
```

```
double calc_phi( double t, double c0, double s0);
double calc_ht( double t, double c0, double s0);
double calc_gt( double t, double s0, struct data_rt data );
```

Sous-routines

```
// Predict Retweet activity using TiDeH:
// Sub-rootins: Memory kernels
#include "TiDeH.h"

// Memory kernel: phi_t
double calc_phi( double t, double c0, double s0);

double calc_gt( double t, double s0, struct data_rt data )
{
  int i; double c0= 1.0/s0/ (1- 1.0/Th), g_t= 0;

  for (i=0; i< data.num; i++)
    { g_t += data.n_i[i]* calc_phi( t-data.t_sp[i], c0, s0); }
  return g_t;
}

double calc_ht( double t, double c0, double s0)
{
  double h_t;

  if ( t <= 0 ) { h_t= 0; }
  else if ( t < s0 ) { h_t= c0* t; }
  else { h_t= c0* ( s0- pow( s0, 1-Th)* ( pow( s0, Th)- pow(t, Th) )/Th
     ↪ ); }

  return h_t;
}


double calc_phi( double t, double c0, double s0)
{
```

```
  double phi_t;

  if ( t < 0 ) { phi_t= 0; }
  else if ( t < s0 ) { phi_t= c0; }
  else { phi_t= c0* pow( t/s0, Th-1); }

  return phi_t;
}
```

G.2 Estimation de l'infectiosité

```
// Estimate TiDeH parameters from Twitter data:
// V.160101 by Ryota Kobayashi - Modified by Pierre Petit
#include "TiDeH.h"

// Cnt. #Retweet in a window
// compte le nombre de retweet dans une time window
int cnt_rt( double t_st, double t_en, struct data_rt data );

// Estimate p(t) from data by moving window.
double est_pt( double t_st, double t_en, struct data_rt data, double s0
   ↪ );


int main(int argc, char *argv[])
{
  if( argc != 5 )
    { printf("Usage:./a.out␣*f_in␣$s0␣␣$T0␣(h)␣f_est_mv\n"); exit(1); }

  FILE *f_in, *f_est_mv;
  f_in = fopen( argv[1], "r");
  if ( f_in == NULL) { printf("Cannot␣open␣$f_in\n"); exit(1); }
  f_est_mv = fopen( argv[5], "w");
  if ( f_est_mv == NULL) { printf("Cannot␣open␣$f_est_mv\n"); exit(1); }

  int i;
```

```
int num;
double t_0;
struct data_rt data;

fscanf( f_in, "%d␣%lf", &num, &t_0);
data.t_sp= calloc( num, sizeof(double) );
data.n_i = calloc( num, sizeof(int) );
data.num= num;

for (i=0; i<num; i++)
  { fscanf( f_in, "%lf␣␣%d␣", &data.t_sp[i], &data.n_i[i]); }

int dt= 1;
int max= atoi(argv[4]);
int *n_d;
double *p_t;
double *x;
double s0= atof(argv[3]);
double t_st, t_en;
n_d= calloc( max-WIN+1, sizeof(int) );
p_t= calloc( max-WIN+1, sizeof(double) );
x= calloc( max-WIN+1, sizeof(double) );


for (i=0; i< max-WIN+1; i++)
  {
    t_st= dt*i; t_en= dt*i+ WIN;

    x[i]= (t_st+t_en)/ 48.0;
    n_d[i]= cnt_rt( t_st, t_en, data);
    p_t[i]= est_pt( t_st, t_en, data, s0);
    fprintf ( f_est_mv, "%f,␣%f\n",t_st, p_t[i]);

  }

free(data.t_sp); free(data.n_i); free(p_t);
fclose(f_in); fclose(f_est_mv);
return 0;
```

```
}

int cnt_rt( double t_st, double t_en, struct data_rt data )
{
  int k= 0, n_rt= 0;

  while ( data.t_sp[k]< t_en && k< data.num ) // k-th event
    { // Cnt. #Events in a window
      if ( k> 0 && data.t_sp[k] > t_st ) { n_rt++; }
      k++;
    }

  return n_rt;
}


// Estimate p(t) from data by moving window.
double est_pt( double t_st, double t_en, struct data_rt data, double s0
   ↪ )
{
  int k= 0, n_rt= 0;
  double c0= 1.0/s0/ (1- 1.0/Th);
  double h1, h2, par, sum= 0;

  while ( data.t_sp[k]< t_en && k< data.num ) // k-th event
    {
      // h_t= \int_0^t \lam_s ds
      h2= calc_ht( (double)(t_en- data.t_sp[k]), c0, s0 );
      h1= calc_ht( (double)(t_st- data.t_sp[k]), c0, s0 );
      // sum: sum_k int lambda(s) ds
      sum += data.n_i[k]* (h2- h1);

      // Cnt. #Events in a window
      if ( k> 0 && data.t_sp[k] > t_st ) { n_rt++; }
      k++;
    }

  if ( n_rt== 0 ) { par= 0; }
```

```
  else { par= n_rt/sum; }

  return par;
}
```

G.3 Prédiction d'activité

```
// Predict retweet activity using TiDeH.
#include "TiDeH.h"

// Solve Integral Equation for activity:
void solv_sc_lambda( int max, double t_st, struct data_rt data, double
   ↪ ave_n, double* par, double s0, double lam[] );

// Num. of Event between t in [t_1, t_2]
int cnt_event( int t_1, int t_2, int num, struct data_rt data);
// Calculate mean number of follwers:
double calc_ave_n( struct data_rt data );
// Calc. p(t) (Infactious rate)
double calc_pT( double t, double* par );

double nb_params;


int main(int argc, char *argv[])
{
  if( argc != 7 )
    { printf("Usage:./a.out␣*f_in␣*f_par␣*f_pred␣$t_st␣(h)␣$s0␣(h)␣
       ↪ $w_bin␣(h)\n"); exit(1); }

  FILE *f_in, *f_pred, *f_par;
  f_in = fopen( argv[1], "r");
  if ( f_in == NULL) { printf("Cannot␣open␣$f_in␣(pred)␣\n"); exit(1); }
  f_par = fopen( argv[2], "r");
  if ( f_par == NULL) { printf("Cannot␣open␣$f_par\n"); exit(1); }
  f_pred = fopen( argv[3], "w");
```

```
if ( f_pred == NULL) { printf("Cannot␣open␣$f_pred\n"); exit(1); }

int i, j, num;
double t_0, t_en= 168, t_st= atof(argv[4]);
struct data_rt data;

fscanf( f_in, "%d␣%lf", &num, &t_0); // Read #Event
data.n_i = calloc( num, sizeof(int) ); data.num= 0;
data.t_sp= calloc( num, sizeof(double) );

for (i=0; i<num+1; i++) // Read event time and #follwers
  {
    fscanf( f_in, "%lf␣%d␣", &data.t_sp[i], &data.n_i[i]);
    if ( data.t_sp[i]< t_st ) data.num++;
  }
// Average Num. of Degree
double ave_n= calc_ave_n(data);

// Read parameters
int max= (t_en-t_st)/DT;
double s0= atof(argv[5]), *lam;

fscanf( f_par, "%lf", &nb_params);
double* par;
par=calloc(nb_params,(sizeof(double)));

  for (i=0; i<nb_params; i++) // Read event time and #follwers
  {
    fscanf( f_par, "%lf␣", &par[i]);
  }

lam= calloc( max, sizeof(double) );

// Predict future activity based on TiDeH
solv_sc_lambda( max, t_st, data, ave_n, par, s0, lam );


int n_bin, n_dat, n_p;
```

```
  double err= 0, sum, w_bin= atof(argv[6]);
  n_bin= (int)(w_bin/DT); n_p= (int)( (t_en- t_st)/w_bin);

  for (i=0; i< n_p; i++) // Show Activity
    {
      n_dat= cnt_event( t_st+i*w_bin, t_st+(i+1)*w_bin, num, data );

      sum= 0;
      for (j=0; j<n_bin; j++) sum += lam[i*n_bin+j];
      sum= sum/n_bin;

      err += fabs( n_dat- w_bin* sum);
      fprintf( f_pred, "%f␣%d␣%f\n", t_st+i*w_bin, n_dat, w_bin* sum );
    }

  free(data.t_sp); free(data.n_i); free(lam); free(par);
  fclose(f_in); fclose(f_pred); fclose(f_par);
  return 0;
}


void solv_sc_lambda( int max, double t_st, struct data_rt data, double
   ↪ ave_n, double* par, double s0, double lam[] )
{
  int i, j;
  double alpha_t, c_t, c0, conv, p_T, t_i;

  c0= 1.0/s0/ (1- 1.0/Th);
  t_i= t_st;
  p_T= calc_pT( t_i, par ); // t_i: (day)
  lam[0]= p_T* calc_gt( t_st, s0, data ); // t_st (hour)

  for (i=1; i<max; i++)
    {
      t_i= (t_st+ i*DT)/1.0;
      p_T= calc_pT( t_i, par );

      alpha_t= ave_n* p_T* DT;
```

```
      c_t= 1.0/ (1- alpha_t* c0/2.0);
      conv= calc_phi( i*DT, c0, s0)* lam[0]/2.0;

      for (j=1; j<i; j++) // Calc. lambda
        { conv += calc_phi( (i-j)*DT, c0, s0)* lam[j]; }

      lam[i]=c_t* ( p_T* calc_gt( t_st+i*DT, s0, data)+ alpha_t*conv) ;
    }
}


int cnt_event( int t_1, int t_2, int num, struct data_rt data)
{
  int k1= 0, k2= 0;

  while ( data.t_sp[k1] < t_2 && k1< num+1 )
    {
      if ( data.t_sp[k1] < t_1 ) k2++;
      k1++;
    }
  return k1-k2;
}


double calc_pT( double t, double* par )
{
  double p_T;
  int i;
  double temp;
  p_T= par[0];
  for(i=1;i<nb_params;i++){
    temp=2*M_PI*i*t;
    p_T+=par[i]*cos(temp/200);
  }
  return p_T;
}
```

```
double calc_ave_n( struct data_rt data )
{ // Calculate mean number of follwers:
  int i; double sum= 0;
  if ( data.num == 1 ) { return 0; }
  else
    {
      for (i=1; i< data.num; i++) // Check up to data.num-1
        { sum += data.n_i[i]; } // sum:sum n_i[i]

      return sum/(data.num-1);
    }
}
```